2002

Integration and visualization of rapid prototyping and reverse engineering.

Tingzhou. Wu
University of Windsor

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

UMI®

Integration and Visualization of Rapid Prototyping and Reverse Engineering

By

Tingzhou Wu

A Thesis
Submitted to the Faculty of Graduate Studies and Research
through Industrial and Manufacturing Systems Engineering
in Partial Fulfillment of the Requirements for
the Degree of Master of Applied Science at the
University of Windsor

Windsor, Ontario, Canada

2001

ABSTRACT

Reverse engineering is an emerging technology that generates engineering geometric design data from existing parts. Rapid prototyping is a relatively new manufacturing process which refers to a class of additive layer-based manufacturing technologies, as opposed to traditional material removal processes. The reverse engineering process consists of the following steps: point data acquisition, noise filtering, data reduction, data arrangement, data registration, data segmentation, curve/surface fitting, and 3D surface model generation. The surface-modeling task from point cloud data, however, is a challenging task, which generally is not automated and requires a significant amount time and skill. It is reported that surface modeling tasks account for over 90% of reverse engineering time (Schoene, 1997).

Rapid prototyping, where a part is built gradually by adding materials layer-by-layer, is fully automatic and it offers many advantages over traditional manufacturing processes. Rapid prototyping process planning includes: part orientation, support structure design, slicing the part into 2D contours, path planning, and process parameters selection.

Integrating reverse engineering and rapid prototyping technologies can facilitate the process of product development. Data transfer is critical in integrating reverse engineering and rapid prototyping process cycle. It must be efficient and free from errors or ambiguity.

The processing of data clouds, from the laser scanner, was investigated in this thesis. Three integration approaches ((i) 3-D model direct slicing. (ii) Mesh generation from unstructured data cloud (Sun, 2001). (iii) Slice contour file generation directly from data cloud (Lee,2001).) have been implemented. The advantages and disadvantages of approaches are examined through case studies.

A visual simulation system has been developed for simulating rapid prototyping process - FDM (Fused Deposition Modeling). The visual simulation technique simulates the part prior to fabricating the physical model. Careful examination of the virtual prototype before the actual fabrication can help minimize unwanted design iterations. The visualization capability of the system is demonstrated by comparing virtual prototypes with corresponding physical prototypes.

ACKNOWLEDGEMENTS

I would like to extend sincere appreciation to my advisor Professor Waguih H. ElMaraghy for affording me guidance and support throughout my M.A.Sc. program. I would also like to express my deepest gratitude to my other committee members, Dr. Hoda A. ElMaraghy, for her rigorous examination, and Dr. Xiaobu Yuan, for his review, of my thesis.

I would like to thank everyone who helped me during my stay in the Intelligent Manufacturing Systems Lab, namely, Mr. Diaa F. Elkott, Dr. Tarek ElMekkawy, Mr. Xiaoyong (Ted) Yang, Mr. Zhentao Wang, Mr. David Faccenda, Ms. Jill Urbanic, Ms. Omayma Nada, Mr. Ayman Youssef, Mr. Amr Shabaka and Mr. Marvin Zhong.

I would like to thank the Industrial and Manufacturing Systems Engineering Program secretary, Ms. Jacquie Mummery and department technical support Mr. Ram Barakat for their help.

I wish to thank my parents for their support, understanding and encouragement. Finally, I must express my utmost gratitude to my wife Mei Yong for her love and assistance, which is the core of all that I have accomplished.

NOMENCLATURE

CAD	Computer Aided Design
RE	Reverse Engineering
RP	Rapid Prototyping
VP	Virtual Prototyping
FDM	Fused Deposition Modeling, one of RP techniques.
SLA	Stereolithography, one of RP techniques.
SLS	Selective Laser Sintering, one of RP techniques.
Catalyst	Catalyst is a front-end processing software which prepares the file for building on Prodigy.
Prodigy	Prodigy is an office modeling system using FDM technique from *Stratasys*.
Sense8	Sense8 product line is a real-time interactive-3D Software Development Tools from *Engineering Animation, Inc.* .
OpenGL	OpenGL (Open Graphics Library) is a software interface that allows the programmer to create 2D and 3D graphics images.
DeskArtes	DeskArtes Expert Series is an expanding family of software tools for checking and fixing 3D data from *DeskArtes*.
ACIS	ACIS 3D Geometric Modeler (ACIS) is a geometric modeling software engine from *Spatial*. Software developers build 3D modeling applications on top of this modeling engine.
IGES	IGES (Initial Graphics Exchange Specification) is an international standard that defines a neutral file format for representation of geometric, topological, and annotation data.

STL STL (Stereolithography) file is the de-facto standard CAD representation for Rapid Prototyping. It was established by *3D Systems* in the late 80s. The STL format of a CAD model is a faceted surface representation, i.e. a list of the triangular surfaces with no adjacency information.

CMB CMB is a toolpath file outputted from Catalyst used to drive Prodigy.

SGM SGM is a contour file outputted from Catalyst.

TABLE OF CONTENT

TABLE OF FIGURES

CHAPTER 1

INTRODUCTION

This chapter presents the reverse engineering and rapid prototyping concepts used in this work. Section 1.1 discusses four topics required to understand the reverse engineering process: part digitization, data segmentation, curve/surface fitting and CAD model creation methodologies. Section 2.1 reviews concepts used in rapid prototyping: techniques' classification, data transfer, accuracy issues and planning techniques. Section 1.3 presents integration methods of reverse engineering and rapid prototyping.

1.1 REVERSE ENGINEERING

Reverse Engineering is a process by which a complex CAD model of an existing part or product is created from a set of measurements produced by various sensors. The created model must describe all of the part's or product's relevant characteristics such as geometry, material property, tolerances, etc. In some cases, where a CAD model of the part already exists, the Reverse Engineering process can also be defined as an update of the part CAD model to reflect the dimensional changes that occurred during prototyping.

The reverse engineering procedure can be characterized by the flowchart in Figure 1.1.

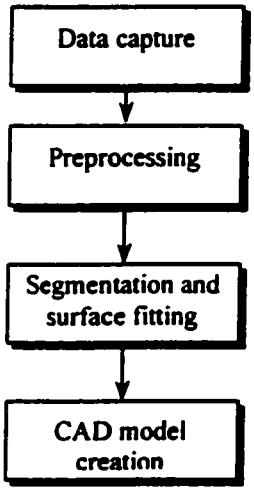


Figure 1.1. Basic phase of reverse engineering (Várady, et al. 1997)

1.1.1 Application Areas of Reverse Engineering

There are several application areas of reverse engineering:

- A prototype of the final product has been modeled manually and therefore no CAD model of the prototype exists (For example clay model in automotive industry);
- A product modeled with CAD is in prototype stage and the required modifications have been applied to the prototype itself but not to the CAD model;
- Worn or broken parts of which no CAD models exist must be rebuilt;
- A CAD system is introduced in a company and all existing products must be modeled in order to have a fully digital archive;
- As an input to a FEA modeler to simulate physical properties of real parts.

1.1.2 Reverse Engineering Part Digitization Methods

The first step in creating a CAD model for an existing part is part digitization. Digitization is a process of acquiring point coordinates from part surfaces. The result of the digitization process is a cloud of 2-D or 3-D data points stored as an image. Part digitization methods may be classified as non-contact methods and tactile methods.

Non-contact methods are classified as those in which the data acquisition device does not physically touch the part. These methods include optical, acoustic and magnetic methods.

Optical methods are probably the broadest and most popular with relatively fast acquisition rates. These methods include:

Structure lighting: involves projecting patterns of light upon a surface of interest and capturing an image of the resulting pattern as reflected by the surface.

Spot ranging: is used to find the range or the z coordinate of a point of interest. A source of light or ultrasound is projected to the part surface. The light or ultrasound is reflected by the part surface and captured by a detector. The z coordinate can be calculated using the travel time or frequency modulation. The x, y coordinate can be found using vision systems.

Range from focus: the focus distance is used to determine the z coordinate. The x, y coordinate can be found using vision systems.

Stereo scanning: the z coordinate can be found using two cameras to view the part form different views. Triangulation is used to find the range information.

Tactile methods touch a surface using mechanical arms. Sensing devices in the joints of the arm determine the relative coordinate locations. CMM is the most commonly used contact device for extracting the 3-D coordinates from part surfaces. Tactile methods are

among the most robust (less noise, more accurate, more repeatable etc.), but they are also the slowest methods for data acquisition.

1.1.3 Segmentation

The surface of the object can be broken into various component surfaces, which meet along sharp or smooth edges. Some of these will be simple surfaces, such as planar or cylindrical surfaces; others will need to be represented by free-form surfaces. The objective of segmentation of scanned data points is to divide the data points into a finite number of bounded subsets of range data, one for each natural surface, so that each subset contains points sampled from a particular natural surface.

Segmentation generally contains two different approaches, namely edge-based and face-based methods (Várady, et al. 1997).

Edge-based method is one popular approach of a two-stage process, edge detection and linking. This work tries to find boundaries in the point data representing edges between surfaces. This technique attempts to find edge curves in the data and infers the surfaces from the implicit segmentation provided by the edge curves.

The face-base segmentation approach goes in the opposite order and tries to infer connected regions of the points with similar properties as belonging to the same surface. Edges are then obtained by intersection of surfaces or other computation.

Edge-base methods suffer from the following problems. Sensor data, particularly from laser scanners, are often unreliable near sharp edge. Because only points in the vicinity of the edges are used to segment, finding smooth edges is very unreliable. On the other hand face-base technologies work on a large number of points, in principle using all available points. Deciding which point belongs to which surface is natural for such methods. This type

of approach provides the best-fit surface to the points as a by-product. So face-based is preferable rather than edge-base segmentation

1.1.4 Surface Fitting

Surface fitting techniques can be classified into two categories: interpolation and approximation (Chivate, et al. 1995). Surface representation methods, in general, are classified as algebraic and parametric surfaces.

Objects with non-discontinuous surfaces can be modeled accurately by fitting algorithms such as: Least square, Least absolute deviation, Least median of squares fits and Patch methods.

1.1.5 Multiple View Combination – Registration

Registration is the process of bringing geometric entities into proper alignment. It is necessary when an object or parts of an object have to be scanned at different times, or have to match coordinate systems of other scanned data. The basic types of registration are 3-2-1, direct, point, set-to-point set, mixed mode, SPT (Spatial/Translational) and stepwise (Imageware 1998).

1.1.6 Model Creation

Several types of shape representation exist, and they are briefly described below. They have all been proposed or used (in some cases experimentally) as input to RP process planning systems.

1.1.6.1 Surface Modeling

A surface model is generally composed of bounded regions of various kinds of surfaces linked together to form a larger composite surface. The surface types used may include planar and other simple surfaces, and also free-form surfaces such as NURBS (non-uniform rational B-splines). Surface models may be either closed (enclosing a volume) or open. 3-D model defined by surfaces uses a precise mathematical description such as Bezier, B-spline surfaces or non-uniform rational B-spline (NURBS) surfaces. A specialized type of surface model uses only planar surfaces and in this case, the composite surface is represented by mesh of planar polygonal facets. If all the facets are triangular, the model can be written out in what is known as the STL format.

1.1.6.2 Constructive Solid Geometry (CSG)

Constructive Solid Geometry, or CSG for short, is constructed from a few primitives with Boolean operators (i.e., set union, intersection and difference). Thus, a CSG solid can be written as a set of equations and can also be considered a design methodology.

1.1.6.3 Boundary Representation (B-rep)

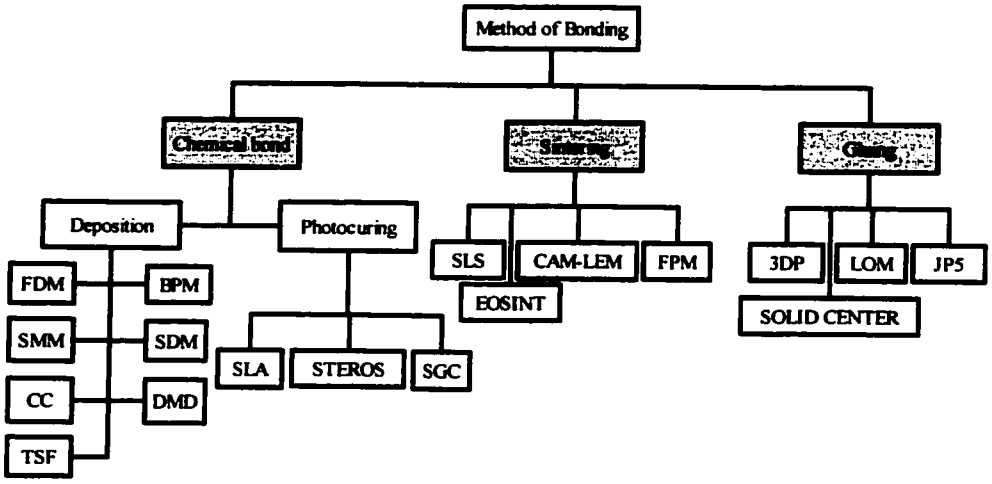
B-Rep models represent a solid indirectly by a representation of its bounding surfaces. A B-Rep solid is represented as a volume contained in a set of surfaces together with topological information which defines the relationships between the surfaces. The boundary of a solid separates points inside from points outside the solid. B-rep models can represent a wide class of objects but data structure is complex, and it requires a large memory space.

1.2 RAPID PROTOTYPING

Rapid prototyping (RP) is a fundamentally different method of fabrication where the object is manufactured from a CAD model by direct numerical control, in 2.5D layers, to create a solid of some predefined shape. A distinct advantage of RP is that the geometric complexity of the part has significantly less impact on the fabrication process than in the case of conventional subtractive manufacturing processes. In addition, RP processes require very little human intervention and setup time, hence, cost-effective production of patterns and moulds with complex surfaces can be achieved. It has been claimed that RP can cut new product cost by up to 70% and the time to market by 90% (Phsm, et al. 1998).

1.2.1 Approaches and Technologies for Rapid Prototyping

Rapid prototyping technologies may be divided into material addition processes and material removal processes. A classification of the material addition processes is show in Figure 1.2. There are few technologies available in material removal processes. Desktop milling system, which is capable of dealing with STL files, belongs to these processes.



	Technology
FDM	Fused Deposition Modeling
CC	Contour Crafting
SMM	Sander Model Maker
DMD	Diredct Metal Deposition
BPM	Ballistic Particle Manufacturing
TSF	Topographic Shell Fabrication
SDM	Shape Deposition Manufacturing

	Technology
SLA	Stereolithgraphy
SGC	Solid Ground Curing
SLS	Selective Laser Sintering
FPM	Freeform Powder Modeling
3DP	3-D Printing
LOM	Laminated Object Manufacturing
JP5	JP5 System

	Technology
CAM-LEM	Computer Aided Manufacturing of Laminated Engineering MAterials
EOSINT	Laser Sintering
STEROS	Laser Photolithography
SOLIDCENTER	Lamination of Paper

Figure 1.2. Classification of rapid prototyping techniques
(Adapted from Kulkarni, et al. 2000)

1.2.2 Data Transfer Formats for Rapid Prototyping

Currently, data describing the boundary of an object is most commonly transferred to the RP process planning system in the form of the STL file. The STL format was originally developed for use with a specific process, stereolithgraphy, but is now widely used by all RP processes. The procedure of data transfer between the CAD and RP system can be characterized by the flowchart in Figure 1.3.

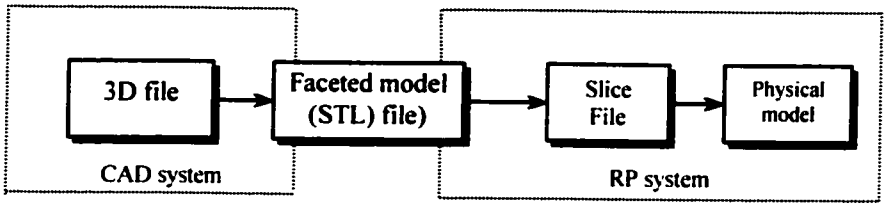


Figure 1.3. Data transfer between the CAD and RP system

1.2.2.1 The STL Format

The STL (Stereo Lithography) format was developed primarily by Albert Consulting Group in 1987 (Marsan, et al. 1998) and is the most widely used *de facto* standard for rapid prototyping systems. The representation used for the 3D part model is a triangulated boundary representation. The format is a list of planar triangular facets, each triangle being defined by three distinct vertices and a normal direction pointing outwards from the interior of the object. The format is specified as both an ASCII (printable character) format as well as a binary format.

Figure 1.4 shows an example extracting from an ASCII STL file. The first line is a description line that starts with the word "solid" in lower case. It then normally contains the file name, author, date etc. The last line should be the keyword "endsolid". The lines between the above contain descriptions of 3 vertex facets including their normals, the ordering of the vertices should comply with the right hand rule.

```
solid  Untitled1
  facet normal  9.86393923E-01  1.64398991E-01  0.00000000E+00
    outer loop
      vertex  9.73762280E-01  7.40301994E-01  1.35078953E+00
      vertex  1.00078931E+00  5.78139828E-01  1.35078953E+00
      vertex  1.00078931E+00  5.78139828E-01  3.50789527E-01
    endloop
  endfacet
 ...

endsolid  Untitled1
```

Figure 1.4. STL ASCII format example

1.2.2.2 Other Formats

Over the last decade, significant advances have been made in all aspects of RP technology. New materials and process capabilities have expanded the application domain of RP to the manufacture of tooling, and to the production of one-of or short run functional parts. Consequently, the nature of the information transmitted to the RP process planning system has become more critical. Drawbacks of the STL format provide the motivation for developing alternative input formats for RP process planning systems.

The following subsections describe some alternatives to the STL format that have been proposed.

1.2.2.3 RPI File Format

The RPI (Rensselaer Polytechnic Institute) format proposed by Rock, et al. (1991) describes both faceted boundary representations and CSG models defined in terms of Boolean operations on primitive volumes. The format allows the specification of 3D transformations, multiple instancing of model elements, and process parameter values. It

defines a set of entities in terms of schemas, and the data is logically subdivided into records composed of fields. A record is an instance of a schema-defined entity, and record fields have specified variable types (integer, float, double, string, boolean and n-bit binary). In the simplest case, this format defines vertices and faces. An indexed list is created of all vertices occurring in the boundary of the object, and the face list then directly references the vertices by their indices in the vertex list.

1.2.2.4 STH – Surface Triangle Hinted format

The STH format (Marsan at al 1998) uses a triangulated boundary representation, with flexible rules for efficient storage of vertex and connectivity information. The representation used by STH is similar to that of STL, but the chosen format requires less storage, and the provision of connectivity information allows the use of more efficient slicing algorithms.

1.2.2.5 STEP – Standard for the Exchange of Product Data

STEP (Kemmerer, et al. 1999) is an international standard for the exchange of product life-cycle data, where life-cycle is broadly interpreted to include design, analysis, manufacture, maintenance and disposal. Several research groups have suggested using STEP for the transfer of data to RP process planning systems. The STEP format can transmit exact geometry and process information, both outside the capability of STH, or STL. STEP has additional advantage that it can handle material and tolerance data.

1.2.3 Accuracy Issues of Rapid Prototyping

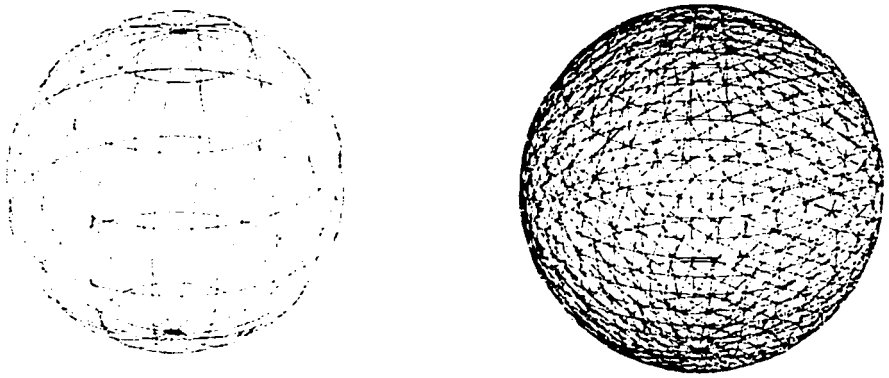
Rapid prototyping systems are controlled by commands translated from the CAD representation and accuracy is core issue in RP industry. It is therefore important to

understand where errors stem from and how to minimize or eliminate these errors. Following section describes accuracy issues associated with rapid prototyping.

1.2.3.1 Tessellation Generation Errors

The tessellation process is translation of a free-form surface to a cell representation consisting of triangles.

Approximation: A major problem with the triangulated boundary representation is that curved surface can only be approximated by triangular facets shown Figure 1.5. A large number of facets give high approximation accuracy, but result in an



(a) (b)

Figure 1.5. (a) B-Rep of a sphere (b) STL approximation of a sphere surface

extremely large amount of data. A small number of facets allow less data but lacks accuracy.

Flipped normals: Facet normals may be flipped and thus be inconsistent with the orientation of other facets. Flipped surface normals may occur when surfaces are improperly defined in the original representation or due to incorrect conversion algorithms.

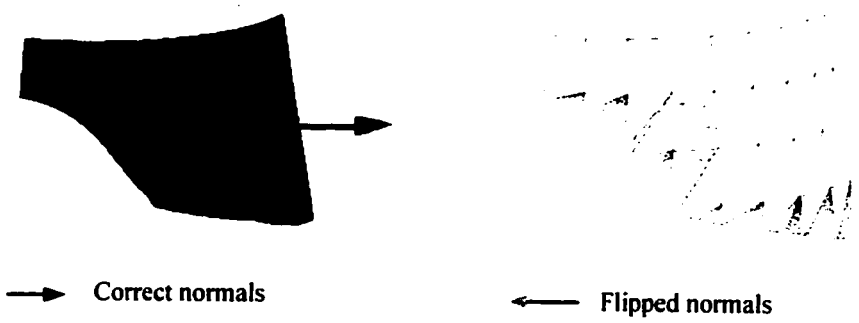
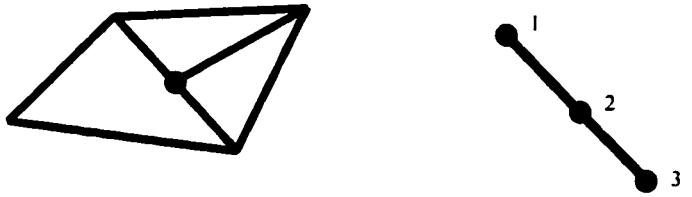


Figure 1.6. Flipped normal in some facets

Degenerate facets: Facets may be degenerate, having zero area and missing surface normal. There are two kinds of facet degeneracy. One is topological degeneracy which two or more vertices of a facet coincide. Another one is geometric degeneracy which all the vertices of a facet are distinct shown in Figure 1.7.



(a) Geometric Degeneracy (b) Topological Degeneracy

Figure 1.7. Degeneracy in STL file

Gaps (closure): This error occurs in translation. Round-off errors cause one point to be at multiple locations at the same time. Thus when triangles are formed, a thin hole is present in the finished model. This type of error is illustrated in Figure 1.8.

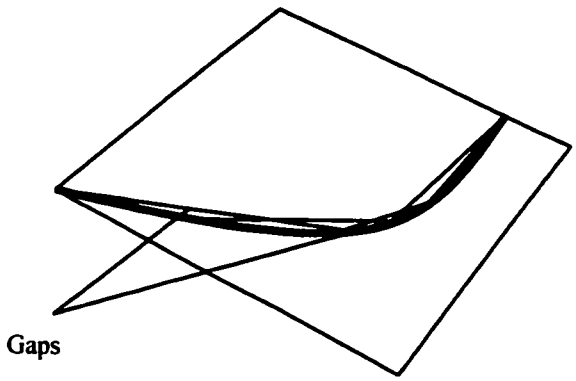


Figure 1.8. Gap due to missing facets

1.2.4 Planning Techniques in Rapid Prototyping

In most rapid prototyping processes, a tool is used to add material to the part. In general, the tool moves while the work-piece is held fixed. In order to drive a RP machine, we need tool paths and process parameter setting.

Process planning is performed to generate the tool paths and process parameters for an object that is to be built by a particular RP process. The steps required are: part orientation, support structure design, slicing the part into 2D contours, path planning, and process parameter selection. Typical RP process planning tasks are shown in Figure 1.9

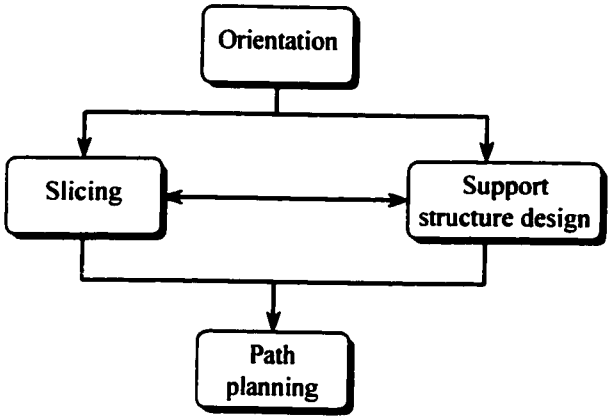


Figure 1.9. General events for RP process planning (Marsan, et al. 1998)

1.2.4.1 Orientation

The build direction of a part will affect the time needed to build the part, its material properties, surface quality and need for support structures. Thus before the part is sliced, an orientation must be determined that is optimal when judged by criteria important to the designer.

1.2.4.2 Supports

RP support structures are used for a variety of reasons, including supporting overhangs, maintaining stability of the part so it does not tip over, supporting large flat walls, preventing excessive shrinkage, supporting components initially disconnected from the main part, and supporting slanted walls. Part orientation and support structure determination are coupled problems, since some orientations of a part will require support structures while others may not.

1.2.4.3 Slicing

Slicing involves computing intersection curves of the part model with a set of parallel planes whose common normals lie in the build direction. A single slice of a faceted model consists of one or more planar polygonal contours, and represents the approximate cross-section of the part at a particular height. The slices may be either uniform or non-uniform vertical spacing, and their geometry is the input to the path planning stage.

1.2.4.4 Path planning

Path planning determines the rapid prototyping tool paths used to build each layer of the part. Rapid prototyping processes are divided into two major types: those in which an entire layer of material is added at once, and those in which each layer is laid down incrementally.

1.3 INTEGRATION OF REVERSE ENGINEERING AND RAPID ENGINEERING

Integrating RE and RP can facilitate the process of new product development. Both technologies use 3D CAD models. In RE, 3D CAD models are created from physical parts, whereas in RP, 3D CAD models are utilized to make physical parts. RE data can be linked to RP in three different paths as shown in Figure 1.10. In the first path, a 3D surface model is created from data clouds and then converted to an STL file. The STL file is then sliced to generate a series of layers for RP fabrication. A STL file is directly created from point cloud in the second path. This path bypasses the creation of surface models. The third path goes directly from point clouds to an RP slice file.

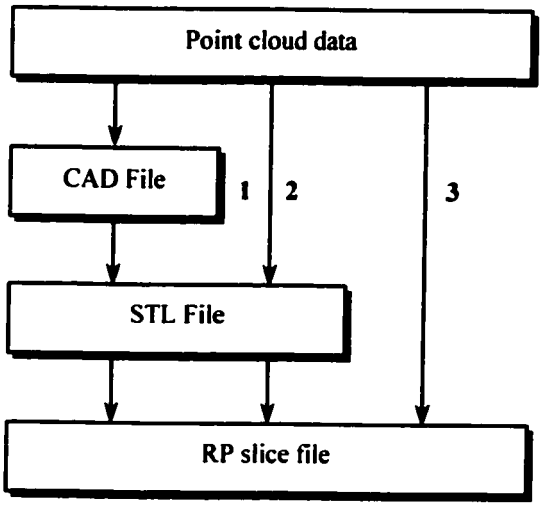


Figure 1.10. Interfacing modes between RE and RP (Lee, et al. 2000)

1.4 THESIS ORGANIZATION

This thesis is divided into seven chapters:

Chapter one presents the background of this thesis, and overviews related techniques.

Chapter two is a literature survey on the current works that relative to reverse engineering, rapid prototyping and integration approaches of reverse engineering and rapid prototyping. It also presents the objective and motivation of this thesis.

Chapter three gives a general description of tools, which were used in this thesis.

Chapter four presents three approaches to integrate reverse engineering and rapid prototyping.

Chapter five presents three approaches' implementation and experimental results as well as comparison among three approaches.

Chapter six presents a visual simulation tool developed in this thesis. The tool can be used to simulate FDM build process and verify the tool path of FDM.

Chapter seven draws the conclusions and gives recommendations for future work.

CHAPTER 2

LITERATURE REVIEW

This chapter reviews some of the important and relevant aspects of reverse engineering and rapid prototyping. Section one presents a survey of the recent literature related to reverse engineering. Rapid prototyping is reviewed in section two. In the third section, recent research pertaining to bridging reverse engineering and rapid prototyping is introduced. Section four presents the objectives of this thesis.

2.1 OVERVIEW OF REVERSE ENGINEERING

Reverse engineering is accomplished in three steps, part digitizing, feature extraction, and CAD modeling (Motavalli 1998). Part digitization is accomplished by variety of contact and non-contact digitizers. Feature extraction is normally achieved by segmenting the digitized data and capturing surface features such as edges. Part modeling is accomplished by fitting a variety of surfaces to the segmented data points.

2.1.1 Data Digitization

The manual digitizing process for a complex free-form surface work-piece is often time-consuming and tedious. Some commercial digitization systems automate the process by

scanning the surface across a rectangular path with a fixed scanning pitch. However, this method tends to pick up redundant points in relatively flat surface regions, and lose the critical points in surface regions with high curvature. Some fast 3D digitizers can scan dense measurement data in a short time, but the scanning result is not guaranteed when strict measure accuracy is required. Furthermore the digitized geometric location and the distributed density of points often do not meet the surface geometric trend. To solve these problems, some approaches were proposed by various researchers. Chen, et al. (1997) proposed an approach for generating free-form surface with a CMM and a touch trigger probe. The approach is divided into two stages: the planning of boundary digitization and the adaptive model-based digitizing process. The planning of boundary digitization defines the measured surface region by recording the necessary points on the surface boundary by manual measurement. The adaptive model-based digitizing process explores points of the interior of exploration regions along a collision-free measuring path. The main idea of the digitizing strategy is to digitize the surface points following the surface local curvature trend.

2.1.2 Feature Extraction – Segmentation

Sarkar (1991) reported an edge detection algorithm to identify the boundaries from the 3-D data points measured on a CMM. Owen (1994) developed a top–down and semi-automatic approach for the segmentation of 3-D points measured on a CMM. It was realized by extracting 3-D points that are close to a given approximate surface in terms of both distance and orientation.

Yang, et al. (1999) introduced a segmentation technique for extracting edges, and partitioning the 3D measured point data based on the location of the boundaries. The procedure begins with the identification of edge points. An edge-based approach is

developed on the basis of local geometry. A parametric quadric surface approximation method is used to estimate the local surface curvature properties. Edge points are identified as the curvature extremes, and zero crossings, which are found from the estimated surface curvatures. After edge points are identified, edge-neighborhood chain-coding algorithm is used for forming boundary curves. The original point set is then broken down into subsets.

2.1.3 Model Creation

For the purpose of solid model reconstruction, the criteria for representation technique are fitting techniques, availability of intersection algorithms, constraint management and the ability to extend the fitted surfaces beyond measured data. A general review of the model representation is provided by Várady's (Várady, et al. 1997) survey paper. Another useful source is given by Chivate, et al. (1995).

In geometric modeling, surfaces are represented by either polyhedral or curved surface approximation. Curved surface approximation methods may be divided into two types: algebraic and parametric. Algebraic surfaces are the ones where the surfaces are approximated using a polynomial equation in the form $f(x, y, z) = 0$.

Vafaeesefat (1998) presented an algorithm to fit sculptured surfaces to a large number of data points obtained by laser scanners. A method based on dual Kriging has been proposed to select the minimum number of knots among existing knots. It avoids the need to fit a surface to a large number of data points.

2.2 OVERVIEW OF RAPID PROTOTYPING

2.2.1 Rapid Prototyping Technologies and Systems

Pham, et al. (1997) presented an overview of the RP technologies and comments on their strengths and weaknesses.

Bradley (2001) presented a system suitable for the three-dimensional (3D) digitization and computer modeling of objects intended for manufacture via computer numerically controlled (CNC) machining or rapid prototyping and tooling systems. The hardware sub-system is comprised of a 3D machine vision sensor integrated with a CNC machine tool. The software sub-system is comprised of modules capable of modeling cloud data using a triangular mesh.

Chen, et al. (2001) developed a robotic machining system with layer based algorithms to build large part models. A model represented as a StereoLithography (STL) or StereoLithography Contour (SLC) file is machined layer by layer. The stock adaptive layer thickness is determined based on a visibility pyramid concept. Each stock layer is in turn machined by a number of machining layers. The cutting tool, robotic arms and the rotary fixtures are calibrated automatically by a machine vision system.

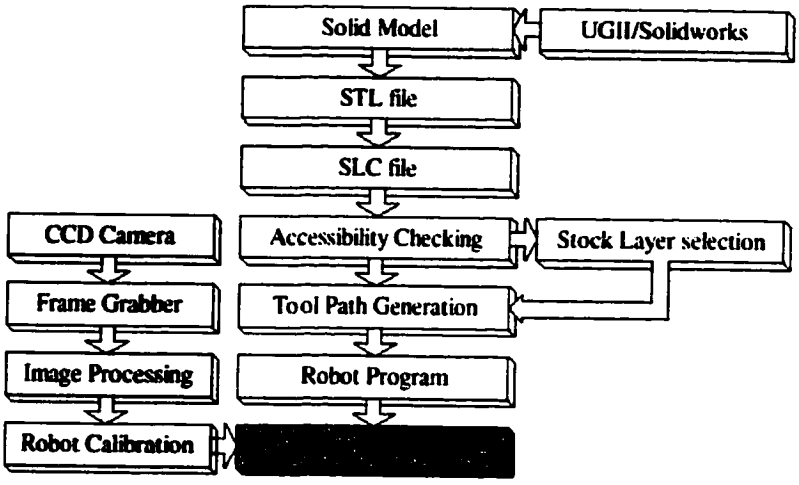


Figure 2.1. Architecture of the robot machining system.

2.2.2 Process Planning Technologies in Rapid Prototyping

In RP, the main research issues stem from the desire to avoid the shortcomings of RP, especially in accuracy and efficiency. Four tasks – slicing, orientation determination, support and path planning – have been identified as constituting the process planning tasks in RP. Kulkarni, et al. (2000) made a good review of process planning techniques in RP. West, et al. (2001) presented a process planning to aid stereolithography user to select appropriate values of process variables. The method use a balance of objectives specified by geometric tolerance, surface finishes, and part build time. Given these objects, values of six process variables are chosen to best achieve the balance of objectives. The process variables include part orientation, layer thickness, and four recoat variables (z-level wait time, sweep period, hatch overcure, and fill overcure). The process planning method is adapted from multi-objective optimization and utilizes empirical data, analytical models, and heuristics to quantitate process variables to the objectives.

2.2.3 Slicing

One of the simplest slicing algorithms for STL files is to intersect all triangles with each z-plane and connect the resulting line segments into closed polygons, one slice at a time. This is the approach that is used by Kirschman, et al. (1992), who also parallelized this algorithm. Model topology is important for a successful model process, unfortunately, STL does not support the definition of model topology. Rock, et al. (1992) built a topological model before slicing and used it to derive connectivity on a slice by marching from one intersected face to the next based on their connectivity. Sabourin, et al. (1997) proposed an approach by subdividing the model space into uniform slabs with the maximum thickness acceptably a RP process. Each of these slabs is further divided into thinner layers so that the cusp height is within a given tolerance. These approaches are based on a STL interfaces.

In order to improve accuracy of slicing, several researchers proposed adaptive slicing algorithms and directly slicing CAD model methods.

2.2.4 Adaptive Slicing

Dolenc, et al. (1994) used cusp height as criterion to adaptively slice STL file. Peak detection was accomplished by looking for degenerate contour.

Kulkarni, et al. (1996) developed a procedure for adaptive slicing of parametrizable algebraic surface using cusp height criterion. Layer thickness is determined from the value of maximum curvature in the vertical direction. Tata, et al. (1998) presented an adaptive slicing algorithm. The algorithm is based on three fundamental concepts: choice of criterion for accommodating complexities of the object, recognition of key characteristic and features (horizontal and vertical surface, pointed edges and end) of the object, and development of a grouping methodology for facets. Four criteria: cusp height, maximum deviation, chord

length and volumetric error per unit length, are identified. Group facets are based on their vertex. Mani, et al. (1999) proposed a region-base adaptive slicing (RAS). In RAS scheme, the model is decomposed into two regions – One region is Adaptive Layer Thickness (ALT) region near the curve surface and is adaptive slice. Another region is the interior Common Interface Layer (CIL) and is sliced at the maximum thickness. The ALT is associated with the critical surfaces and maintains the surface accuracy.

2.2.5 Slicing without Tessellation

As prototyping systems and CAD systems proliferate, many system designers are concerned with developing the means by which 3-D solid or surface data can be transferred to the model-making systems. Generally, most interfaces adhere to the STL format, however, the transfer process is not always complete or "bug-free". Since all prototyping machines require a clean, complete and a valid 3-D model to produce apart of quality. The transfer process proves to be a weak link in the process prototyping. Tessellation is the approximation of an actual model. Hence, tessellated model slicing propagates the errors from CAD into the manufacturing process. Some researchers focused on direct slicing of CAD models to avoid approximation representation. Jamieson, et al. (1995) at Cranfield University used C language directly accessed Parasolid underlying programs and sliced a solid using software call. Then they converted parasolid's slicing contours to CLI (Common Layer Interface), HPGL or SLC (3D Systems contour algorithm). Rajagopalan, et al. (1995) developed an interface between CAD and rapid prototyping systems. The method used intersection routines within I-DEAS (Integrated Design Engineering Analysis Software) to slice the 3D-models. Yan and Gu (1996) explored using ray tracing algorithm to intersect with solid primitives of CSG representation. Chiu, et al. (1998) presented an algorithm for hollowing

out a solid CAD model. The algorithm employs voxel elements to do the hollowing. One-dimensional Boolean operations between the ray representations of the model and voxel elements are carried out. The ray representation of the hollowed model produces the direct slice files as output to the rapid prototyping machine. Ma and He (1999) presented an adaptive slicing algorithm which operates directly upon a NURBs-based CAD surface. A selective hatching strategy was also proposed to reduce the build time by solidifying/depositing kernel regions of the part with the maximum allowable thick layers. In addition, the selective hatching strategy provides an alternative solution to the containment problem.

2.2.6 Data Exchange in RP

Unlike a solid model, a surface model consists of a collection of separate surfaces. The topological information is either absent or insufficiently represented. Without this information, verifying the completeness and correctness of models is difficult. The trimming curves of two neighboring surfaces or their boundaries do not exactly match, even when they result from the same intersection of surfaces. After transforming the surface into STL file, the layer boundaries of the model after slicing might not be contiguous or complete. This results in cross-section filling errors and thus rejected parts. Sheng, et al. (1995) presented an algorithm that builds topological structure for surface models by merging the surface boundaries according to the manifold criteria.

Recently the STL format is the most commonly used interface in RP. But STL suffers from limitations, such as approximation, truncation errors, inconsistency, normal information redundancy etc., other interfaces have being proposed by various researchers. Marsan, et al. (1998) has compared various 3-D and slice formats proposed for layer manufacturing (LM)

and recommend the development of a STEP AP specifically for the purpose of LM process planning. STEP resources already exit for handling LM process planning, including process parameters, tolerance and surface finish requirements. Recently, STEP lacks modeling heterogeneous objects. Jacob, et al. (1999) proposed a alternative file format (LMI) which supports the STL format. The new format eliminated redundant information as in the case of STL format. The format includes topological information and supports precise models by using the edge-base boundary representation.

Instead of finding a new interface, Koc, et al. (2000) presented a method to reconstruct the slicing contour data through biarc curve fitting to improve accuracy of the STL while reducing the necessary data size.

2.2.7 Heterogeneous Objects

Recent developments in the field of structural optimization result in heterogeneous objects in which there is a continuous material variation along with the geometry. A new heterogeneous modeling scheme has been proposed (Kumar 1998) which enhance the capability of existing solid model technologies to capture variations in material in addition to geometry. Bhashyam, et al. (2000) has set up a prototype heterogeneous CAD system incorporating synthesis tools consisting of a library of material composition functions, a library of property estimation rules. Such a system serves as a tool to perform various tasks involved in design, analysis and manufacture of heterogeneous objects.

Chiu, et al. (2000) proposed a scheme for representing multiple material objects in a CAD system. A material tree of the object is combined in the CAD system's data structure. A material list is added into the STL file to suit to the RP fabrication.

2.3 OVERVIEW INTEGRATION OF REVERSE ENGINEERING AND RAPID PROTOTYPING

Schoene and Hoffmann (1997) proposed a triangulation method based on multi-axis digitized data. Multi axis digitizers were carried out on a HURON milling machine fitted with a Heidenhain digitizing system. Since unordered point clouds with undercuts often result in faulty STL files. It is necessary generate interactive tools for handling STL data files. They presented an intelligent algorithm for triangulation based upon point clouds. This algorithm can process point clouds even with undercuts and handle unordered point clouds.

Alciatore, et al. (1996) developed a computer aided sculpting software for reading and formatting digitized data and exporting it to RP. The software imports 3D polygonal mesh in STL, OBJ and DXF formats and then reshapes it. Specifying a wall thickness gives the model volume, prior to exporting an STL file. The method includes a data structure used to store and modify the model information, free-form stretching algorithm, resolution enhancement and decimation techniques, and an algorithm for determining surface geodestics, which allow a user to draw 3D shapes directly on the surface of the model.

Chen, et al. (1999) proposed a method using digitized part data directly construct an optimized STL file. In order to reduce storage space and increase computational efficiency for subsequent processes such as slicing, significant data reduction can be achieved by the users' discretion by deleting triangles in planar and near planar regions. Points around the 'blank region' left by deleted triangle are linked through re-triangulation to form triangular facets obeying STL file rules. To obtain optimized re-triangulation result, a genetic algorithm (GA) is developed and implemented.

Sun, et al. (2001) describes an application of error-based triangulation to very large sets of three-dimensional (3D) data. The algorithm is accomplished from the 3D data points in two steps. Firstly, an initial data thinning is performed to reduce the data set size. Secondly, the triangulation commences utilizing a set of heuristic rules, from a user defined seed point. The triangulation algorithm interrogates the local geometric and topological information inherent in the cloud data points. The spatial filtering parameters are extracted from the cloud data set, by a series of local surface patches, and the required spatial error between the final triangulation and the cloud data. Two procedures are subsequently employed to enhance the mesh: (i) the edges of mesh triangles are adjusted to produce a mesh containing approximately equilateral triangles; and (ii) mesh edges are aligned with the boundaries present on the object to minimize smoothing of naturally occurring features.

2.4 THESIS OBJECTIVES

Though both RE and RP are emerging technologies that have been well developed, little research has been done in integrating the two. This provides the potential to research optimal methods, with respect to process time, accuracy. To this end, several objectives have been defined:

1. To develop techniques and guidelines for choosing methods for integrating reverse engineering and rapid prototyping.
2. To create a conceptual framework for an integrated reverse engineering and rapid prototyping system, which makes use of existing and newly developed techniques.

3. To compare the strengths and weaknesses in integrating reverse engineering and rapid prototyping methods.
4. To develop a visual simulation tool to simulate FDM process.

CHAPTER 3

EXPERIMENTAL TOOLS AND SYSTEM ARCHITECTURE

A number of experimental tools were employed in this work. The major software and hardware resources utilized in this thesis are described in this chapter.

3.1 HARDWARE

3.1.1 DEA Mistral Coordinate Measuring Machine

DEA Mistral is a combination manual/direct computer control (DCC) coordinate measuring machine that provides a variety of measurement and inspection options in a single machine. The DEA Mistral, Shown in Figure 3.1, has a linear positioning accuracy of X and a volumetric accuracy of Y. A Renishaw PH9/TP2 probe was used to collect digitization data. Tutor for Window was used to compile DEA Part Programming Language (DEAPPL).

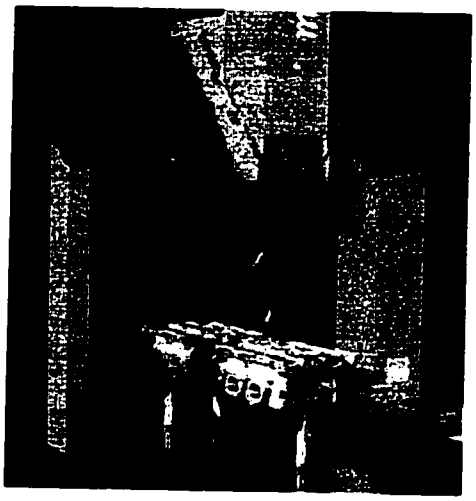


Figure 3.1. DEA Mistral coordinate measuring machine

3.1.2 Hymarc Hyscan 45C Laser Scanner

Hyscan 45(shown in Figure 3.2) is a high performance 3D scanning laser digitizer which rapidly acquires precise XYZ data points from any object. Based on unique and patented *synchronized scanning* technology, Hyscan systems map surface information in

Figure 3.2. Hymarc Hyscan 45C Laser Scanner

a continuous high speed non-contact measurement with ± 0.001" (± 0.025mm) ±3 sigma accuracy, which satisfies demanding applications such as reverse engineering, design, rapid prototyping and inspection.

3.1.3 Stratasys Prodigy Rapid Prototyping System

Prodigy (shown in Figure 3.3) is an office 3D modeling printer using FDM technology. Fused Deposition Modeling (FDM) is a nonlaser-based process, developed in 1988 by Scott Crump. It takes CAD wire frame, surface, or solid models and builds the parts by depositing layers of molten thermoplastic materials.

Prodigy combines the ease, speed and simplicity of 3D printing with the major benefit of ABS functional part durability – all within a networked office environment. Design engineers can now directly print out fully functional and produce their design right at their CAD workstation.

Prodigy is driven by Catalyst, a powerful new front-end processing software. A simple function brings STL files into Catalyst for automatic processing. Running on Windows NT, Catalyst quickly prepares the file for building on Prodigy. Users may select Fine, Standard, or Draft layer thickness based on their needs for feature detail and surface smoothness.

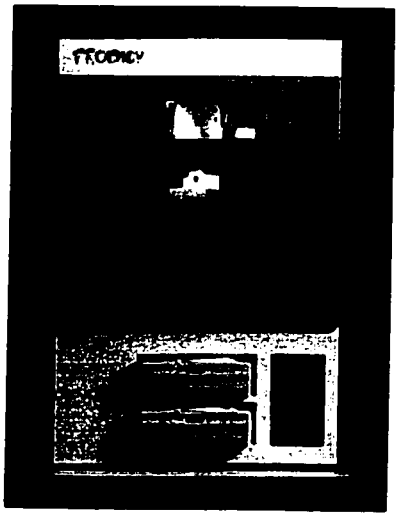


Figure 3.3. Stratasys Prodigy -3D printer

3.2 SOFTWARE

3.2.1 Imageware Surfacer

Imageware Surfacer (Imageware 1999) provides application-driven solutions in the areas of free-form product design, rapid surfacing, high-quality surfacing, reverse engineering, computer aided verification, polygonal modeling, rapid prototyping, surfacing and surface healing. It enables customers to design, accurately build, and fully inspect high-quality, free-form shaped products in less time.

3.2.2 ACIS Toolkit

The ACIS 3D Geometric Modeler (Spatial 2000) is an object-oriented, C++ library. ACIS integrates wire frame, surface, and solid modeling with both manifold and non-manifold topology. It provides software developers with a rich set of geometric operations for the construction and manipulation of complex 3D models.

3.2.3 I-DEAS Master Series

I-DEAS (SDRC 1999) software from SDRC is a scalable, integrated, CAD/CAM/CAE solution designed to support a concurrent engineering methodology. Features include 3D component, assembly and sculptured-surface modeling, kinematic and dynamic mechanism analysis, finite element modeling (FEM), tolerance analysis and toolpaths generation for milling and turning. During this work, application of the software was primarily limited to the construction of solid models from CMM or laser scanned data.

3.3 EXPERIMENT TOOLS ARCHITECTURE

Figure 3.4 demonstrates how hardware and software have been used in the research.

DEA Mistral CMM and Hyscan 45C laser scanner were used to acquire point coordinates from part surfaces. Data clouds were preprocessed by Imageware Surfacer. 3D model creation was performed by I-DEAS. ACIS 3D toolkit and Visual C++ (Microsoft 1998) were utilized to develop integration algorithms. DeskArtes Expert Series (DeskArtes 2001) was employed to repair STL files. STL model preprocessing and RP toolpath generation are carried out by Catalyst (Stratasys 2000). The accuracy comparison is fulfilled by using Imageware Surfacer. Prodigy is used to build prototypes.

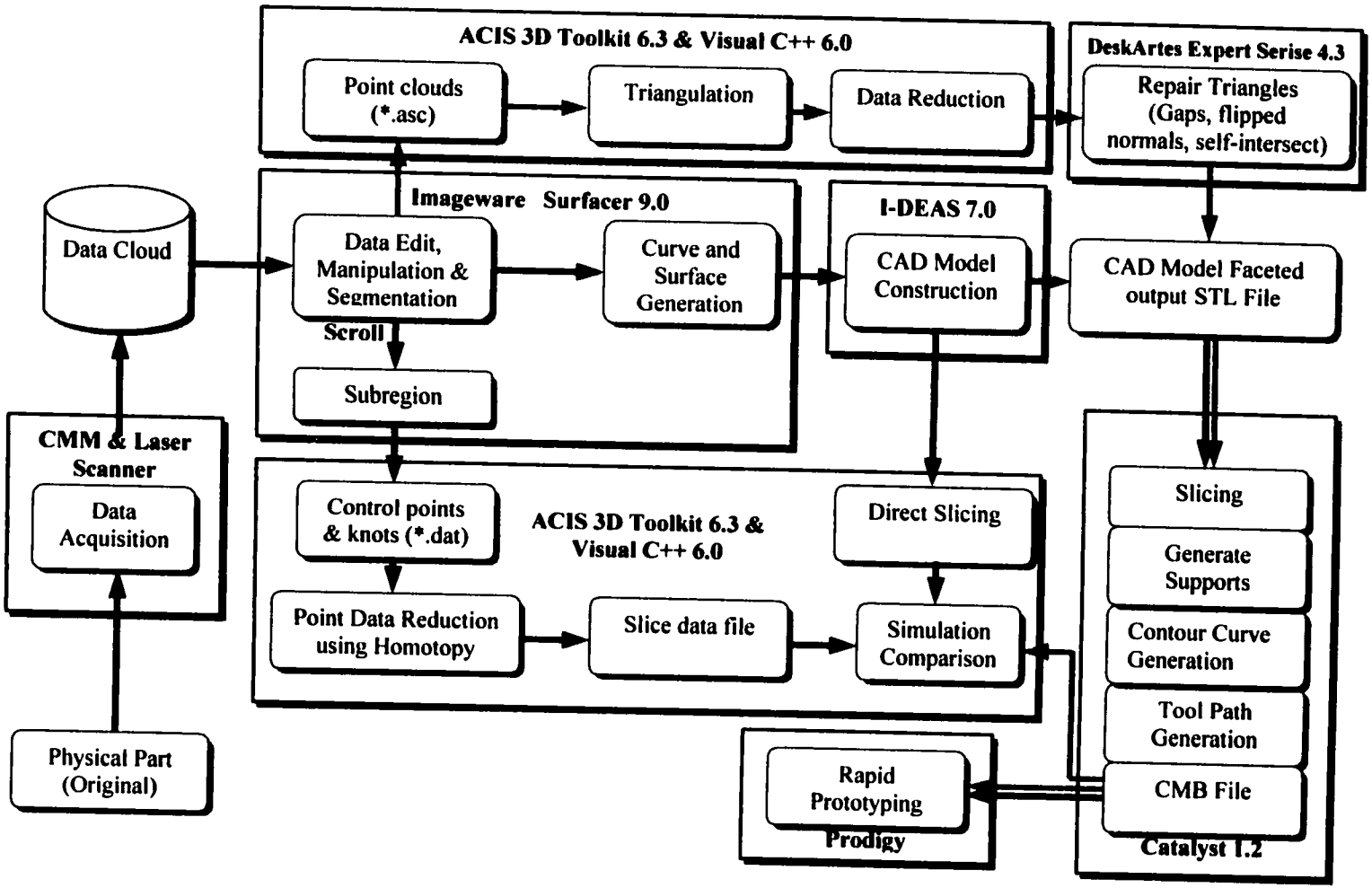


Figure 3.4. System Architecture

CHAPTER 4

INTEGRATION METHODS DESCRIPTION

To compare different approaches to integrate RE and RP, three different methods were proposed in this chapter. Figure 4.1 illustrates the overall processes. In the first approach, a 3D surface model is created from data clouds and then converted to an STL file. The STL file is then sliced to generate a series of layers for RP fabrication. A STL file is directly created from point cloud in the second approach. This approach bypasses the creation of surface models. The third approach goes directly from point clouds to an RP slice file.

4.1 AN APPROACH USING DATA CLOUD TO GENERATE 3-D MODEL THEN SLICING

4.1.1 Data Digitization

A non-contact laser-based range sensor (Hymarc, Hyscan 45C) was used to collect the cloud data from the object surface. The accuracy specification of the cloud data is ±0.025 mm, over a 100-mm depth of field, and a stand-off distance of 100 mm. The sensor scans a laser spot over an object's surface, in a continuous high-speed manner,

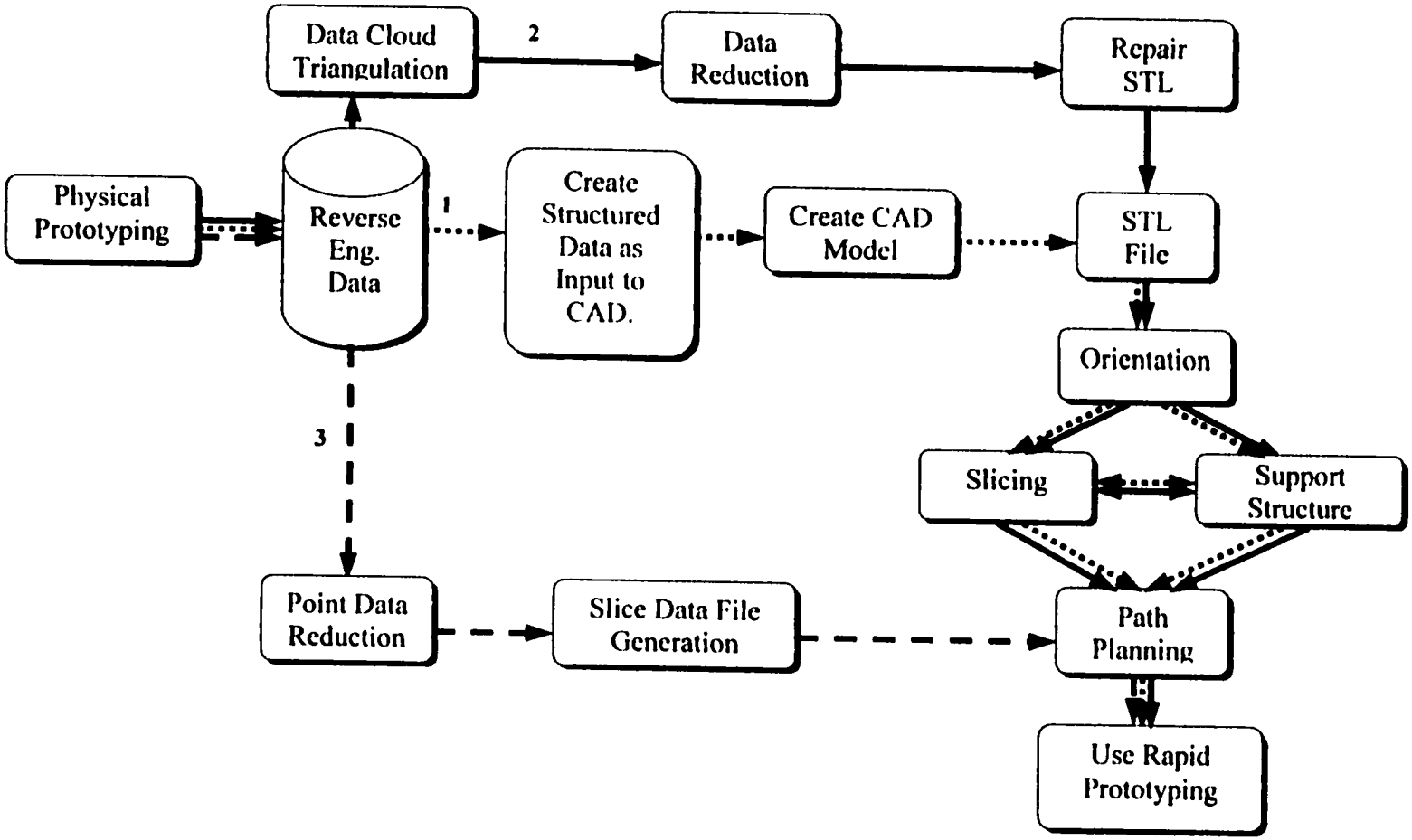


Figure 4.1. Overall flow chart from RE to RP

and gathers data at a maximum rate of 10,000 points per second. The sensor's small field of view and the object's form can often necessitate multiple scanning passes to full digitize the object. A typical digitization process requires the range sensor to be repositioned six-to-eight times, at various positions and viewing angles, to completely sample the surface patches. The resulting cloud data file is a mosaic of range data patches as shown in Figure 4.2, the image of the cloud data collected from the model.

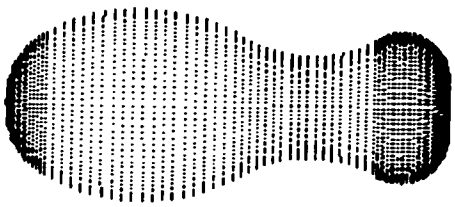


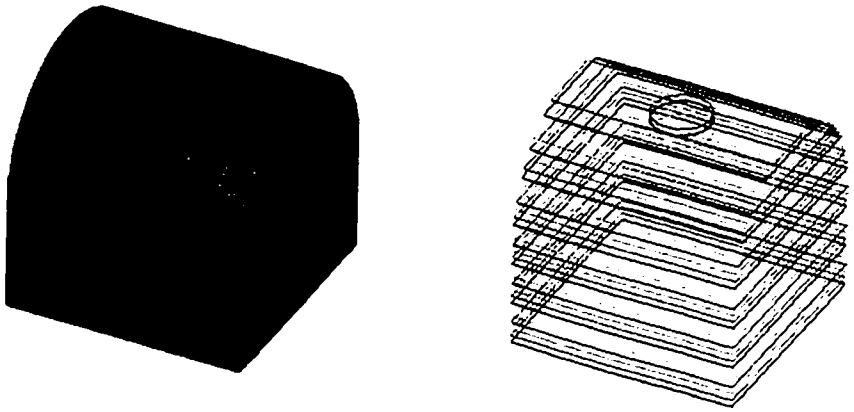
Figure 4.2. Point cloud example: a spin

4.1.2 Direct Slicing

The 3-D model slicing procedure can be regarded as the Boolean intersection of an infinite plane and a part. The operation involves the intersection of a plane defined as a rational B-spline surface with the surfaces that define the model part that is also rational B-spline surfaces. This result in a cross-sectional profile of the part that is composed of a series of smoothly connected rational B-spline curves. The intersection operation can be performed by using ACIS modeler's intersection capabilities. The profile can be processed to generate the boundary data.

4.1.2.1 Uniform Slicing

The simplest slicing procedure used only one parameter, the layer thickness. This is shown in Figure 4.3.



(a) (b)

Figure 4.3. Direct Slicing a Block
(a) Block ACIS model (b) The result of slicing a block.

4.1.2.2 Peaks Handling

Peak features of an object are easily missed when performing slicing (Dolenc, et al. 1993). A peak can be defined as follows. The notation S_i means a slice computed at $z = z_i$. Consider a model with no flat areas. Take a slice S_i and a contour $C \in C(S_i)$ that is not degenerate. If there exists a slice S_i with $z_j < z_i$ such that C becomes degenerate, e.g. it collapses to a point, then the contours in the interval $[z_i\text{-}l, z_i]$ are said to be a peak, where l is the layer thickness. A peak feature may occur at places including corner, a boundary curve, and interior area of a surface.

Thinner layers should be used when peaks are detected. The simplest heuristic procedure is to detect changes in the number of contours in adjacent slices S_i and S_{i-1} , The slice S_{i+1} is discarded, and slicing resumes with alternative, user-defined thickness l_p until S_{i+1} is reached. When the slice S_{i+1} is evaluated and a peak is detected, additional layer is added between S_{i+1} and S_i as shown in Figure 4.4.

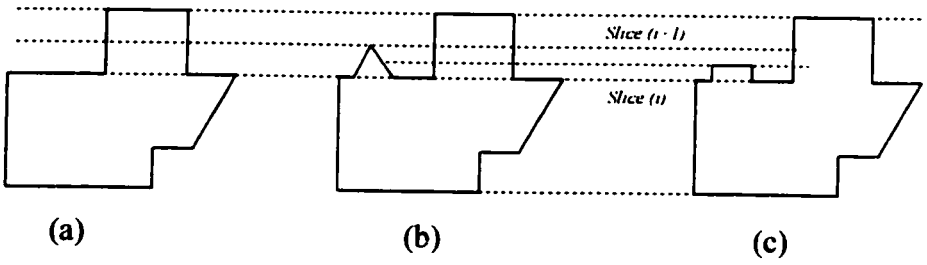


Figure 4.4. (a) Uniform slicing missing peak (b) original model
(c) Using thinner layer thickness at peak

4.1.2.3 Flat Areas Manipulating

Flat areas are areas parallel to the slicing direction. A horizontal slice employing uniform slice thickness cannot take them into account. Identifying the flat areas not only is important to preserve dimensional tolerance of the part but also is useful to simplify and improve the postprocessing for the RP process.

The flat areas can be classified into positive areas and negative areas, depending on their normal directions (see Figure 4.5).

Flat areas are detected by finding planar faces with normal $n = (0,0,1)$ or $n = (0,0,-1)$.

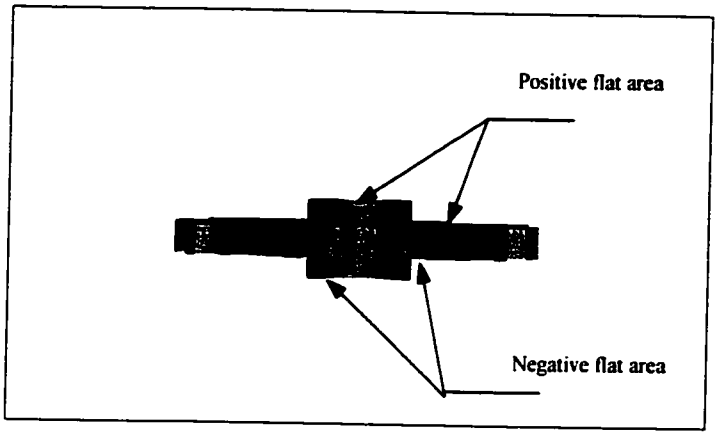


Figure 4.5. Handling flat areas

After flat areas in the part are retrieved, the slicing heights are decided under two constrains: (1) maximum allowed slice thickness determined by machine. (2) Slicing thickness account for flat areas.

Accordingly, from a given slicing heights, the next slicing height is calculate in two steps:

- New slice height = current slice height + standard slicing thickness
- If there is flat area between the current height and the new height, the new slice height will be equal to flat area height.

4.1.2.4 Profile Hatching

When parts are built in layers, the cross section area of each layer has to be defined and filled with a pattern of vectors. The filling process is called hatching. The process of building a part is that a tool traces the contour of the part on the sliced surface followed by cross-hatching. The contour and hatch lines are specified as a set of boundary vector. Hatching is a problem of finding the vectors whose end points are the intersections of the part

profile, which is composed of several rational B-spline curves and straight lines, say the lines y = Constant and x = Constant for X-Y hatching. A cross-sectional contour composed of a series of smoothly connected rational B-spline curves.

Intersection of a straight line and a curve

Denote the straight-line segment by $\mathbf{q}(t) = \mathbf{a} + t\mathbf{b}$, the curve by $\mathbf{p}(u)$, here we assume a B-spline curve and the point of intersection by $\mathbf{r}$. Then, an intersection occurs anywhere $\mathbf{p}(u) = \mathbf{q}(t) = \mathbf{r}$, or (Mortenson 1985)

$$\mathbf{p}(u) - \mathbf{q}(t) = 0 \tag{4.1}$$

A rational B-spline curve can be expressed as following equation (Mortenson 1985)

$$\mathbf{p}(u) = \frac{\sum_{i=0}^{n} N_{i,k}(u) P_i h_i}{\sum_{i=0}^{n} N_{i,k}(u) h_i} \tag{4.2}$$

where P_i are the defining control points in 3-D; $N_{i,k}(u)$ are the B-spline blending functions, h_j is the weight associated with the control point P_i.

The points of intersection can be acquired by solving the following set of equations(Rajagopalan 1995):

$$x = \mathbf{P}_x(u) = \frac{\sum_{i=0}^{n} X_i N_{i,k}(u) h_i}{\sum_{i=0}^{n} N_{i,k}(u) h_i} \tag{4.3}$$

$$y = \mathbf{P}_y(u) = \frac{\sum_{i=0}^{n} Y_i N_{i,k}(u) h_i}{\sum_{i=0}^{n} N_{i,k}(u) h_i} \tag{4.4}$$

$$z = P_z(u) = \frac{\sum_{i=0}^{n} Z_i N_{i,k}(u) h_i}{\sum_{i=0}^{n} N_{i,k}(u) h_i} \tag{4.5}$$

$$f(x,y,z) = 0 \tag{4.6}$$

The equation can be solved by numerical methods. After getting intersection points, next stage involves sorting these points to determine the portions that form the "inside" of the object. The interior of a slice bounded by several contours can be determined by using the odd-winding rule. This rule states that a point is in the interior of the slice if a ray extending from the point intersects the slice's boundary an odd number of times (Figure. 4.6).

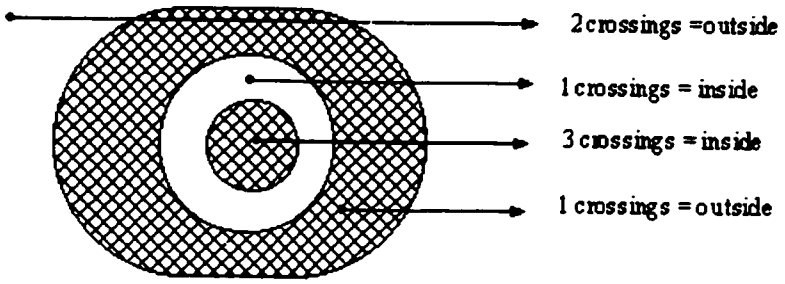


Figure 4.6. Odd-winding rule

4.2 AN APPROACH USING DATA CLOUD TO GENERATE TRIANGULAR MESH (STL FILE) THEN SLICING

4.2.1 Unstructured Cloud Data Processing

The method used in this thesis applies voxel binning (Sun, et al. 2001), to the cloud data, as an initial step. The bin size is first calculated based on a required error tolerance between the original cloud data and the final triangulation.

The overall steps for computing the triangulation are presented below:

- Compute bounding box for the cloud data points.
- Offset the bounding box limits outward by a point coincidence tolerance (TOL). This step resolves conflicts when points lie on cell planes.
- Sample N points from the cloud data file and interpolate a local cubic B-spline surface patch to the N points by means of least squares fitting.
- Calculate the maximum second partial derivatives from the cubic function coefficients. The derivative values are used to calculate the maximum triangle edge length (Ω) for a desired error tolerance ε between the cloud data and triangulation.
- Determine maximum triangle edge length Ω, from the derivative values and ε. Compute the bin cell size (B), corresponding to ε, from the following expression:

$$B \leq \sqrt{\frac{3\varepsilon}{2(M_1 + 2M_2 + M_3)}} \qquad (4.7)$$

where, M1, M2 and M3 are the maximum partial derivative values of the locally interpolated surface.

- Apply voxel-binning data to reduce the cloud data
- Construct the final triangular polyhedral mesh to the thinned data set.

4.2.1.1 Sample Points from the Cloud Data File

A spatial subdivision method is used to sample points from the cloud file. The method divides cloud points into a grid of small squares or cells. Through this method, each point of cloud data will dropped into appropriate cell. Points in each cell will be used to fit

surface patch. The size of the grid is controlled by a *cell occupancy target M* -a number representing the desired average number in points per cell. The grid size *m* expresses as $m = \sqrt{\frac{n}{M}}$. Where data cloud contains *n* points. Experiment (Sun, et al. 2001) revealed that a small set of 24 to 32 points was an effective trade-off between computation time and sufficient data to form a reasonable interpolation to the underlying surface form. So the range of *M* can be around 24 to 32. (Figure 4.7 shown a gird)

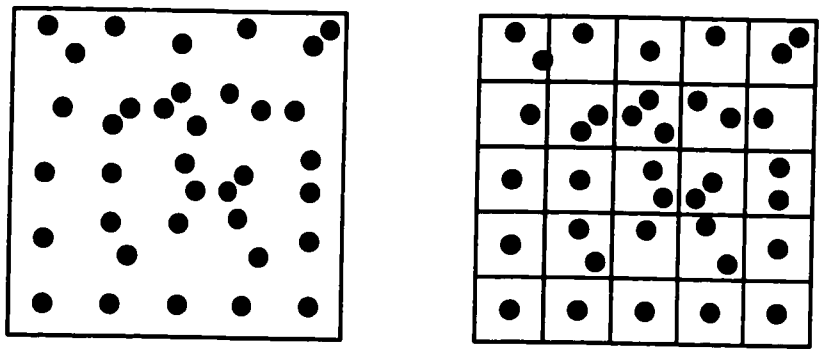


Figure 4.7. A grid method

4.2.1.2 Patched Surface Fitting

A piecewise polynomial surface is a grid of parametric polynomial patches which can be considered as (Sheng, et al. 1992):

$$\mathbf{S(u,v)} = \sum_{j=0}^{m}\sum_{i=0}^{n} a_{ij} u^i v^j \quad u \in [u^1, u^2],\ v \in [v^1, v^2] \qquad (4.8)$$

Where a_{ij} are the vertices of the defining polyhedron, and m & n are the curve degrees in the *u* & *v* directions respectively. Surfaces are defined by the bounded rectangle of parameter *u* and *v*.

4.2.1.3 Triangle-Size Bound Calculation

The problem can be formulated as follow (shown Figure 4.8): given a parametric patch of a C^2 surface **S** and an arbitrary triangle **T** with its vertices on the surface and triangle within the same parametric bound. If the distance is smaller than a user-defined tolerance, the patch is said to be sufficiently flat. The deviation between the triangular facet and the surface patch satisfies (Sheng, et al. 1992)

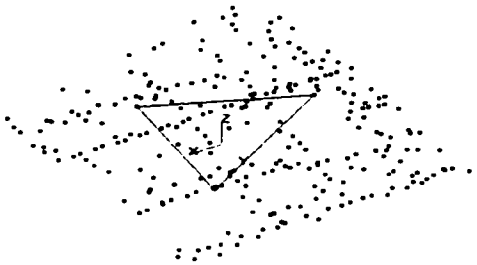


Figure 4.8. Triangle in bound

$$\sup_{(u,v)\in T}\|S(u,v)-T(u,v)\| \le \frac{2}{9}\Omega^2(M_1+2M_2+M_3) \qquad (4.9)$$

$$M_1 = \sup_{(u,v)\in T}\left\|\frac{\partial^2 S(u,v)}{\partial u^2}\right\|$$

$$M_2 = \sup_{(u,v)\in T}\left\|\frac{\partial^2 S(u,v)}{\partial u \partial v}\right\|$$

$$M_3 = \sup_{(u,v)\in T}\left\|\frac{\partial^2 S(u,v)}{\partial v^2}\right\|$$

Following this formula, one can easily determine the maximum allowed length Ω for the given tolerance ε.

$$\Omega \leq 3\sqrt{\frac{\varepsilon}{2(M_1 + 2M_2 + M_3)}} \tag{4.10}$$

4.2.1.4 Bin Size Calculation

The maximum triangle edge length determines the maximum bin size. It is assumed that the surface could pass through two neighboring bins in the following ways: (a) through bins sharing a common face; (b) obliquely pass through two neighboring cells (i.e. at 45°) that share a common edge; and (c) through bins that share a point. Therefore, the maximum linking distance l_{max} between two points in neighboring cells is given by:

$$B = \frac{\sqrt{3}}{3} l_{\max} \tag{4.11}$$

The maximum length of a triangular facet edge is equivalent to the maximum distance l_{max}. The maximum bin size is derived as:

$$B \leq \sqrt{\frac{3\varepsilon}{2(M_1 + 2M_2 + M_3)}} \tag{4.12}$$

4.2.1.5 Data Reduction

The data thinning process proceeds using the computed voxel bin size. Typically, multiple data points will fall within a single voxel and, therefore, each bin is examined to determine which point should be retained. The retained point is closest to the voxel's centre, thereby reducing the cloud data to a more uniformly spaced set.

4.2.1.6 Triangulation

A triangular structure corresponding to a given set of points is not unique. Among the many possible triangulations, Delaunay triangulation (Laszlo, 1996) is considered to

construct triangles. The Delaunay triangulation of a point set is a collection of edges satisfying an "empty circle" property: for each edge we can find a circle containing the edge's endpoints but not containing any other points.

4.2.2 Slicing

4.2.2.1 Edge/Plane Intersections

Methods of computing edge/plane intersections are fundamental to the slicing procedure. The slicing planes used to slice a solid model are by definition horizontal. Rock and Wozny (1991) show how this greatly simplifies the edge/plane intersection process. If one edge vertex is above the plane and the other point is below, then the edge intersects the plane.

In the Euclidean coordinate system ($\vec{i},\vec{j},\vec{k}$), the equation of a horizontal plane at a given height h is given by

$$z = h \tag{4.13}$$

If the edge is defined by two vertices $\mathbf{p} = (x_p, y_p, z_p)$ and $\mathbf{q} = (x_q, y_q, z_q)$ as described by the parameterization

$$\mathbf{p}(t) = \mathbf{p} + t(\mathbf{p}-\mathbf{q}) \tag{4.14}$$

Then the edge/plane intersection can be found at

$$t = \frac{h - z_p}{z_q - z_p} \tag{4.15}$$

4.2.2.2 Contour Orientation Convention

The material in each layer is bound by the set of contours on the corresponding slice. The convention for these contours is that they are directed such that the material always lies to the left of the contours, as viewed in the direction of the contour. Hence, external contours are directed counterclockwise (CCW) and internal contours are directed clockwise (CW)

4.2.2.3 Marching Algorithm for Slicing

Marching algorithm (Rock, et al. 1991) is used to create slice contours. A marching algorithm requires a valid geometric model with knowledge of connectivity between the components of the model to be marched through. Model slice and contour generation of a faceted solid model is greatly simplified if facet connectivity is known. An order contour could be extracted given an initial facet that intersects the slice plane. Each intersecting facet has two intersecting edges. The facet's contour edge is bounded by these two edges' intersection with the slice plane and is connected with two contour edges arising from the intersection of the two other facets adjacent to the two intersecting edges. Hence, the contour edges can be found by marching from intersecting facet to intersecting facet.

The direction of marching is given by $\mathbf{Z} \times \mathbf{N}$, where $\mathbf{Z}$ is the vertical vector and $\mathbf{N}$ is the facet's surface normal. For instance, a facet f_a ($\mathbf{V}_1$, $\mathbf{V}_2$, $\mathbf{V}_3$), with the vertex $\mathbf{V}_1$ above and the vertex $\mathbf{V}_2$ below the slice plane, the next facet will be f_{13} if $\mathbf{V}_3$ is below the plane or f_{23} if $\mathbf{V}_3$ is above the slice plane as shown in Figures 4.9 and 4.10.

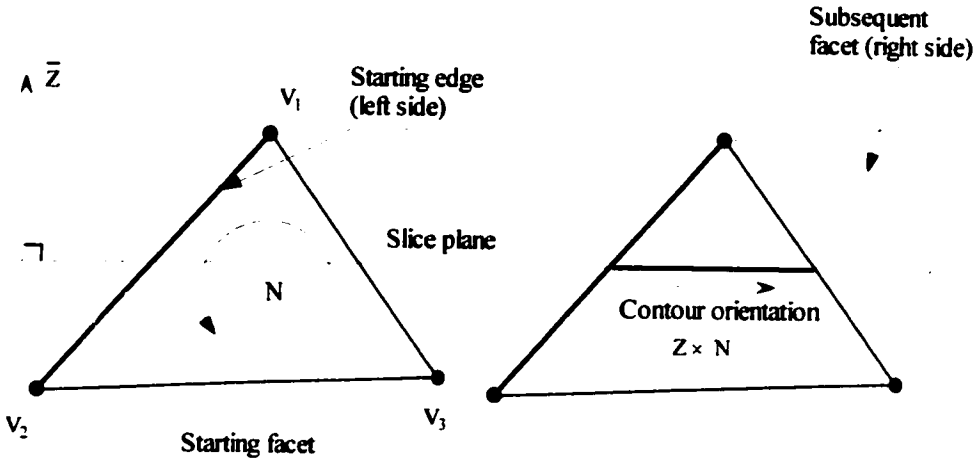


Figure 4.9. The marching is given by **Z** × **N**

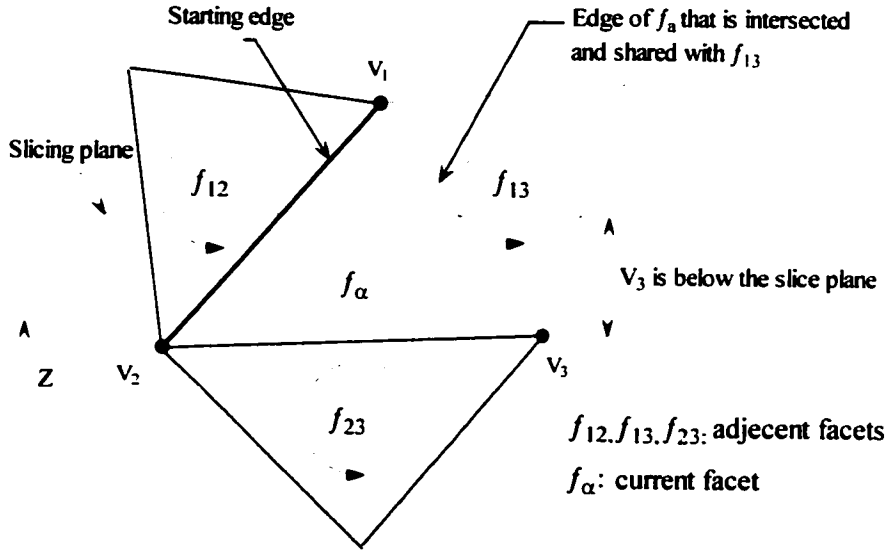


Figure 4.10. Given a staring facet f_a and a staring edge ($v1$,$v2$), the next facet to intersect is determined by the relative position of the vertex $v3$ to the slicing plane. Here, since $v3$ is below the plane, f_{13} is the next facet to be intersected by the slice plane

4.3 AN APPROACH USING DATA CLOUD TO DIRECTLY GENERATE SLICE FILE

4.3.1 Point Data Arrangement

Point data are arranged as a series of cross-sectional contours. In order to construct cross-sectional contours, the point clouds are divided by a series of parallel planes, giving a constant space between two adjacent parallel planes. The point data between two adjacent parallel planes are rearranged by projecting them on the corresponding cross-sectional planes.

4.3.2 Curvature Information Extraction

In RP, part is built by sweeping each cross-section vertically by the height of a layer and the results in zero vertical curvature for each slice. The curvature data on the cross-sectional planes is kept, but ones along the vertical direction are approximated. It is crucial to keep critical features such as edges and peak points along the z-axis. It can be achieved by keeping cross-sectional contours at z-heights where sharp curvature changes occur.

4.3.2.1 Generation of vertical cross-sections

The vertical cross-sectional planes can be generated along the y-axis, x-axis, or around the z-axis (as shown Figure 4.11). One of methods should be chosen according to the geometry of the part.

(a) x-axis cross-sections (b) y-axis cross-sections

Figure 4.11. Methods for generating vertical cross-section

4.3.2.2 Determination and Extraction of Curvature Changing Points

A set of 2-D point data is acquired for each vertical cross-sectional plane. For each point in 2-D point data set, the angular deviation can be calculated using the vectors created from the consecutive points as shown in Figure 4.12.

Figure 4.12. Angular deviation method for extracting points

4.3.2.3 Calculation of Point Extraction Ratio

The extracting data from all vertical cross-sections are gathered to recreate the point model. The edges and features where curvature significantly changed are detected and the overall geometry of the part is identified.

$$r_{zi} = \frac{N_{zi}}{T_{zi}} \qquad (0 \le r_{zi} \le 1)$$

where N_{zi}: The number if extracted points at height Z_i; T_{zi}: Total number of points in the cross-section at height Z_i

4.3.3 Subregioning

Dividing point data model into subregions can facilitate the data reduction process. Boundary contours of subregion can be generated by the horizontal cross-sections located at z-heights where the point extraction ratio is greater than threshold value. The purpose of subregioning is to ensure accurate fabrication of edges and critical features. The boundary contours can be as reference data during the data reduction process.

4.3.4 Point Data Reduction

For point data reduction, the points on a horizontal cross-sectional contour have to be compared with those of the neighboring boundary cross-sections. If the deviation is with the given tolerance, the points in that contour can be eliminated; otherwise, they need to be kept. A homotopy method is introduced to compare the point data.

Homotopy Method

Intermediate contours between the boundary contours can be generated using a homotopy. A homotopy is defined as follows (Drakos 1994).

Let I denote the unit interval [0, 1].

Let f, g: $X \rightarrow Y$ be two maps, where X and Y are topological spaces. Then f and g are homotopic if there exists a continuous function

$$H: X \times I \rightarrow Y$$

such that H (x, 0) = f (x) and H (x, 1) = g (x). If $A \subseteq X$ is a subspace such that f and g agree on A, then H is called a homotopy of f and g relative to A or rel A if $H(a, t) = f(a)$ for all $a \subseteq A$

and $t \subseteq I$. When H is defined by $H(x, t) = (1 - t) \cdot f(x) + t \cdot g(x)$, it is called a straight-line homotopy.

The homotopy can be used in different ways. The homotopy is used to generate intermediate contours. A surface that connects two boundary contours can be represented as well by the locus of homotopy that transforms one contour to the other. Figure 4.13 illustrates an example using homotopy.

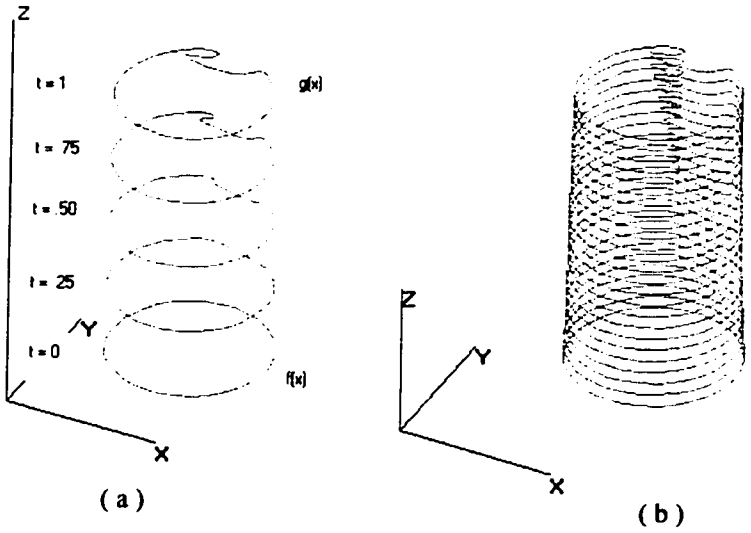


Figure 4.13. Homotopy example: (a) Straight-line homotopy, (b) Generation of intermediate contours

4.3.5 Elimination of Point Data at Intermediate Contours

Boundary contour is generated by point data using B-spine representation. A mid-contour between the consecutive boundary contours can be acquired using homotopy. Comparing mid-contour with corresponding cross-section point data, when the deviation of the each point from the mid-contour is within a given tolerance, the cross-section point data will be removed. Otherwise if the deviation of the each point from the mid-contour is greater

than a given tolerance, the cross-section point data will be kept and the cross-section point data become a boundary contour of new subregions. The process of halving the boundary contours continues until the height of the subregion is smaller than two times of allowable minimum layer thickness. The maximum deviation and the average deviation are used to check the tolerance.

4.3.6 Slice File Generation

The straight-line homotopy is used to generate additional contours between boundary contours. The number of layers in RP fabrication coincides with the number of contours generated.

Slice data file consists of a layer file and a scan file. A layer file contains the contour curves of all layers and a scan file contains the tool path in the contour of each layer.

Figure 4.14 shows the flow chart of the direct contour generation method.

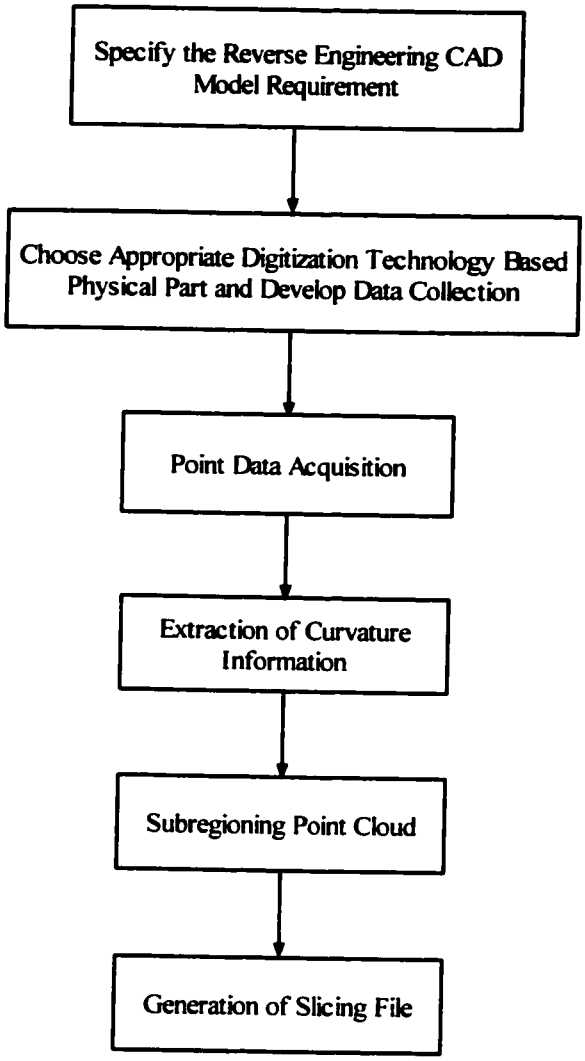


Figure 4.14. Using data cloud to directly generate slice file process

CHAPTER 5

IMPLEMENTATION AND RESULTS

This chapter describes the implementation performed on the integration methods introduced in the previous chapter. It discussed the tools used for the implementation and the results of applying the methods.

The application of the integration methods to selected cases is also introduced. Different physical parts were selected and the integration methods were applied to them. The comparison of accuracy for these methods is demonstrated.

5.1 IMPLEMENTATION OF DIRECT SLICING

The slicing algorithms were developed using ACIS Geometric Modeler and Visual C++ on PC with 128Mb of RAM. The direct slicing algorithm starts with getting a part's bounding box to inquire about the part's highest and lowest points. The slicing direction is the Z direction, so the top is the highest and the bottom is the lowest of z coordinate. Before determining slice thickness, the flat areas in the part are retrieved. Flat areas are usually critical to overall dimensional tolerance of the part. The slicing heights are organized under

two constrains: (1) maximum allowed slice thickness determined by machine. (2) Slicing thickness account for flat areas.

Accordingly, from a given slicing heights, the next slicing height is calculate in two steps:

- New slice height = current slice height + standard slicing thickness
- If there is flat area between the current height and the new height, the new slice height will be equal to flat area height.

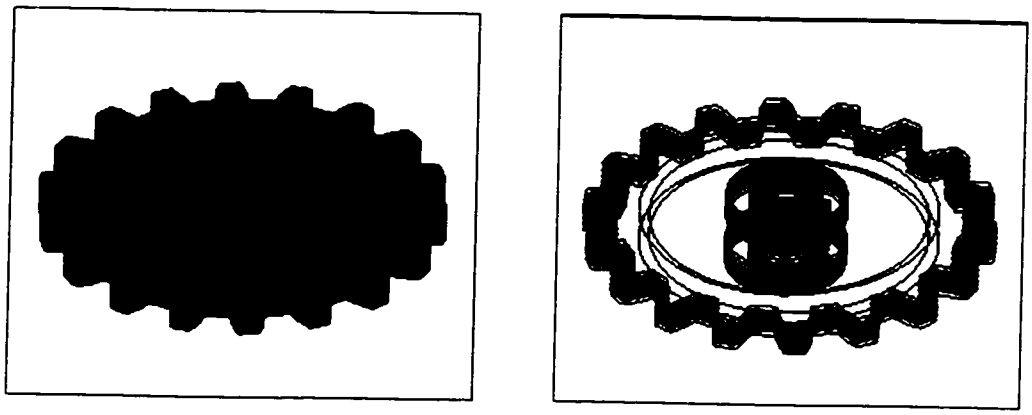
A horizontal plane defined by slice height and normal vector is used to intersect the part. A wire body represented by rational B-spline curves will be generated after operation.

The slice thickness can be set at any value (e.g. 0.15, 0.2, 0.25mm) as input data in the program.

To ensure the usability of the program, a number of tests were carried out. The slicing procedure was test using several different parts.

5.1.1 Example 1: A Gear

In first example, a gear, shown in Figure 5.1 (a), was used to test the usability of the program. The part contains several flat areas and multiple contours in a given cross-section. Figure 5.1 (b) illustrates the result of direct slicing.



(a) (b)

Figure 5.1. (a) CAD model of gear (b) Slicing contours of a gear

Table 5.1. Deviation of STL slicing model from direct slicing model

0.25	0.118	0.058	0.031
2.94	0.145	0.060	0.034
7.60	0.376	0.271	0.064
13.72	0.137	0.137	0.089
17.15	0.148	0.153	0.147
24.01	0.100	0.054	0.039

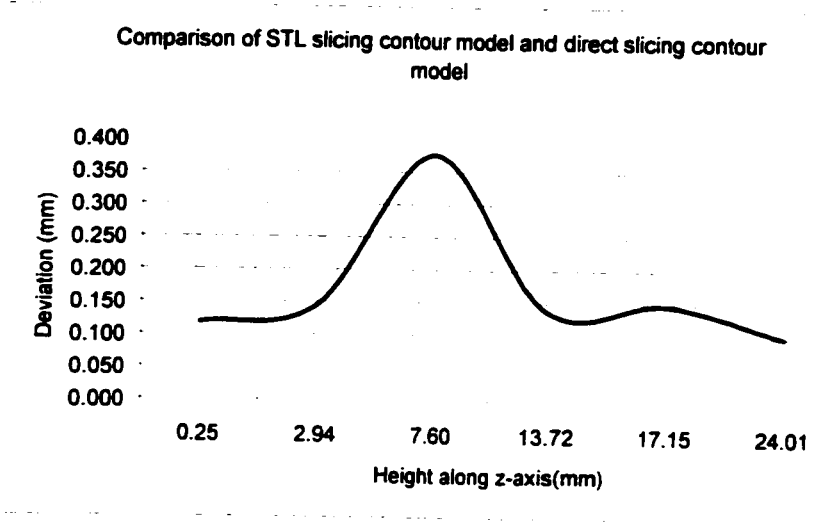


Figure 5.2. Comparison of deviation of STL slicing model from the direct slicing model

5.1.2 Example 2: A Rook

A more complicated CAD model, a rook as shown in Figure 5.3 (a), was tested in second example. Some cross-sections in the model comprise of more than two separate contours. The slicing result of the rook is shown in Figure 5.3 (c)

The direct slicing can be beneficial in terms of file size and cutting out the need to slice a tessellated equivalent model. Accuracy can be enhanced, especially on rounded or tubular design.

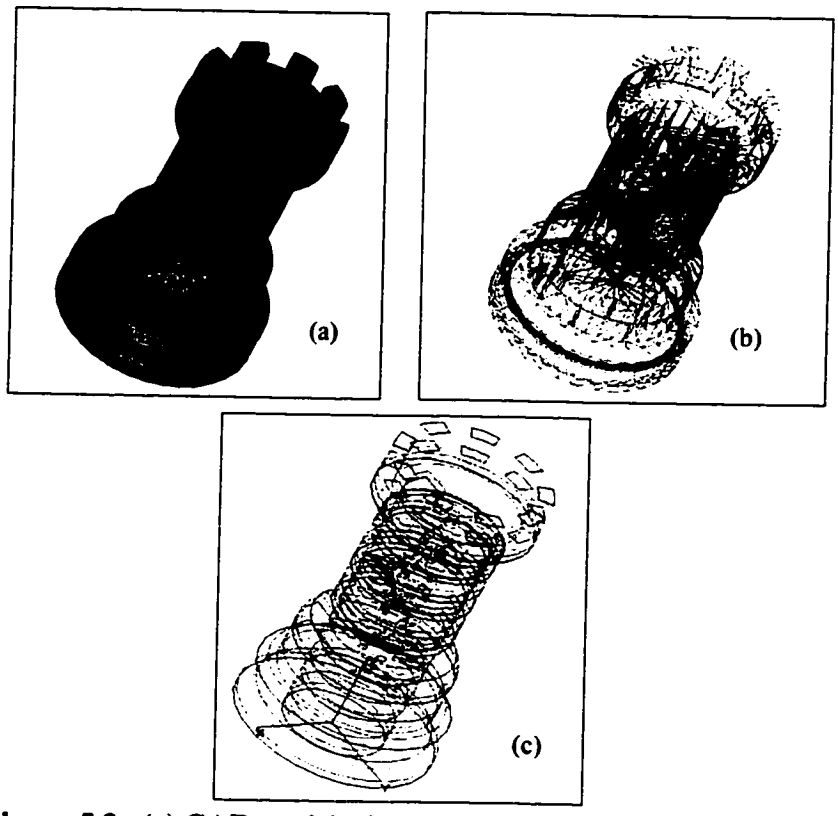


Figure 5.3. (a) CAD model of rook (b) Triangle mesh model of rook
(c) Slicing model of rook

5.2 IMPLEMENTATION OF TRIANGULATION MESH GENERATION FROM UNSTRUCTURED DATA CLOUD

The algorithms in this approach were developed using Visual C++, ACIS toolkit and Surfacer 9.0. Visual C++ and ACIS toolkit were used to develop data cloud processing program. Surfacer 9.0 was utilized to generate polyhedral triangulation meshes and output STL files. 3Data Expert (DeskArte) was used to verify, repair, and modifier STL model.

In the data cloud processing program, a data cloud scanned by laser scanner (Hymarc. Hyscan 45C) is processed. The process starts by extracting the bounding box of the data cloud. A point coincidence tolerance is then used to offset the bounding box. Grid algorithm

divides the data cloud into uniform size patches. Patch size is decided by the number of sample points. Surface patches are then fitted to points in each patch. After the surface patches are fitted, the maximum second partial derivations are calculated from the surface patches. The voxel bin size is computed using maximum second partial derivatives and the specified error tolerance. The bin cell size as is applied to determine which point should be retained in the data cloud. The ASCII file format is selected as an output file format in the data cloud processing program.

Triangular mesh generation is performed in the Surfacer 9.0. The objective of mesh generation is to cover the reduced point set with a single surface mesh of triangular facets. The mesh process is initiated with the section of a seed point. The rest points are sorted by the Euclidean distance from the seed point. The mesh is formed by the seed point and the rest sorted points.

Two examples were tested by using the developed technique.

5.2.1 Example 1: A Light Cover

The first example is a light cover, shown in Figure 5.4, which is scanned by laser scanner (Hyscan 45C). The dimensions of the bounding box of the part are 345.163 × 158.881 × 61.228 mm. The data cloud processing program was applied to the original data cloud occupying 21,809 points. Setting an error tolerance of 0.400 mm, a bin size length of 2.974 mm was decided. 5,179 points were remained after data reduction. A reduced data cloud is shown in Figure 5.5 (a). Using polygon algorithm, the triangular mesh was obtained

as shown in Figure 5.5 (b). An error tolerance of 1.000 was applied to another test and resulted in a bin size of 4.702mm. 2,710 points were remained after data thinning.

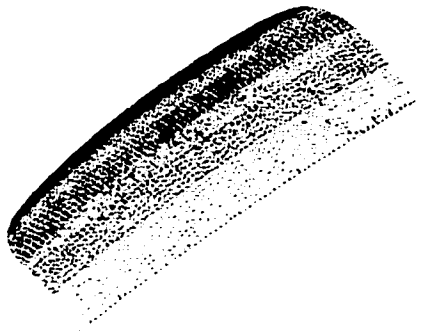


Figure 5.4. Data cloud of a light cover

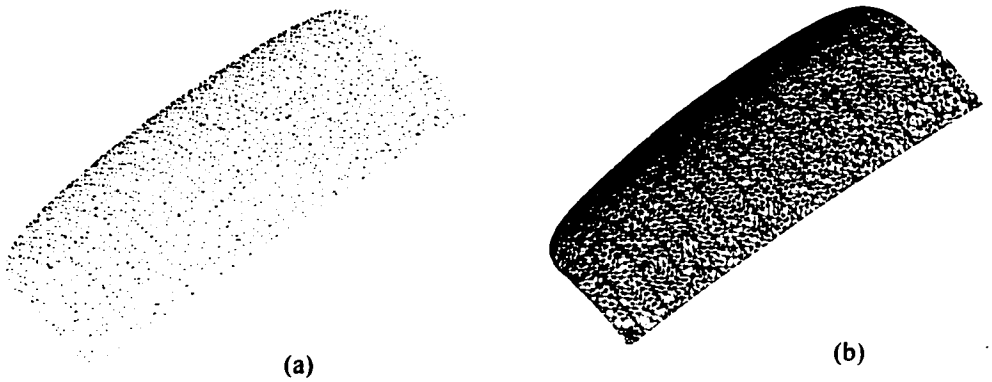


Figure 5.5. (a) Reduced data cloud (b) mesh model using reduced data cloud

5.2.2 Example 2: A Human Face

The second example is a human face (Imageware data sample) as shown in Figure 5.6 (a). The human face is an intricate free-form surface. The surface contains smooth regions connected with sharp feature with high curvature. The original part occupies a volume of

147.000 × 247.500 × 113.000mm. The data cloud processing program was applied to process the data cloud with 9,727 points. The first trial applied an error tolerance of 0.500mm and resulted in bin size of 2.080. 7,120 points were remained after data reduction. The trial of an error tolerance of 1.100mm was performed and resulted in bin size of 3.085mm. 4,016 points were retained before mesh generation as shown in Figure 5.6 (b). Figure 5.6 (c) illustrates the mesh model of the face.

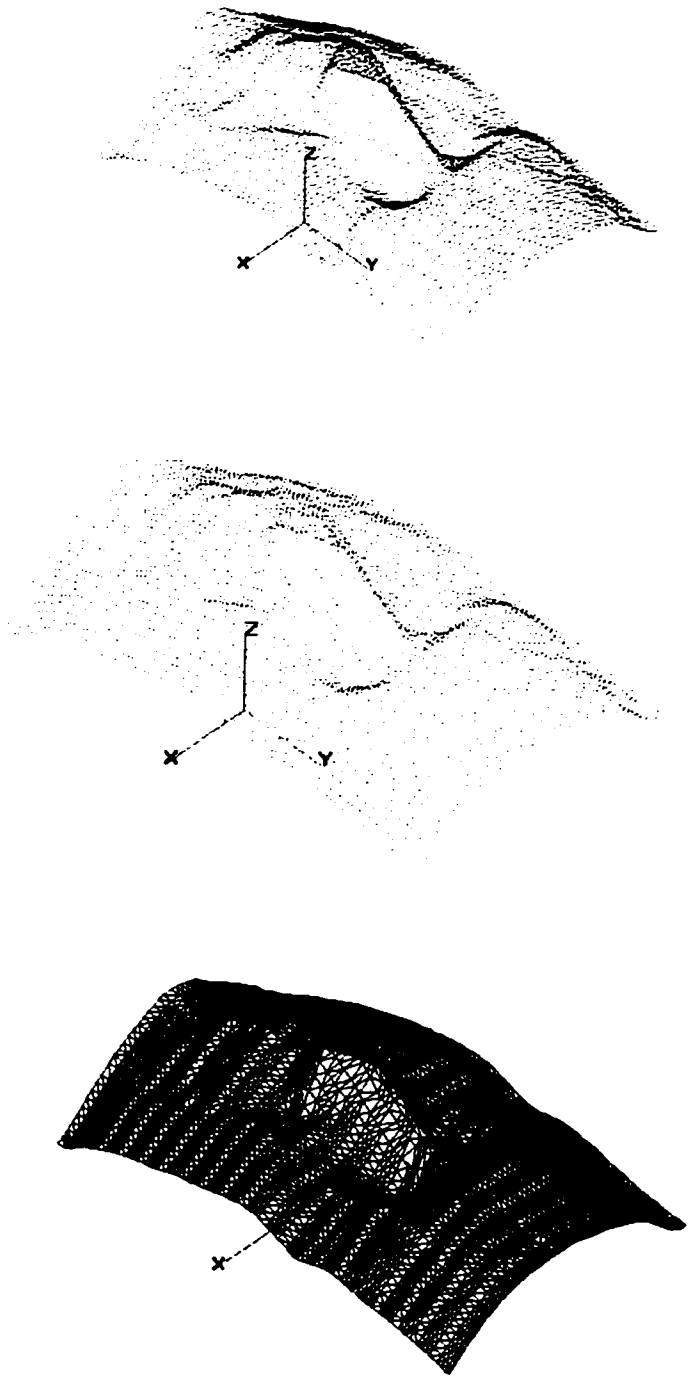


Figure 5.6. (a) Original data cloud (b) Reduced data cloud
(c) Mesh model using reduced data cloud

Table 5.2. Summary of results for each example

	Original number of points	Error Tolerance(mm)	Bin Edge Size (mm)	Number of points after data reduction
Light cover I	21,809	0.400	2.974	5,179
Light cover II	21,809	1.000	4.702	2,710
Face I	9,727	0.500	2.080	7,120
Face II	9,727	1.100	3.085	4,016

5.3 IMPLEMENTATION OF DIRECT CONTOUR GENERATION

The algorithms were developed using Surfacer 9.0, Scroll, the ACIS geometry modeler, and Visual C++ on PC with Windows 98 operating system.

Scoll, a Command Line Language of Surfacer, is designed to make automatic operation for repetitive, interactive tasks. Scoll supports multi-dimensional vectors and associative arrays. Imageware software functions can be implemented as built-in functions in scroll. In this work, Scroll was used to process point data arrangement, extraction of curvature inform, point models sub-region and curves fitting (Code shown in appendix A, output file example shown in appendix B). The flow chart is shown in Figure 5.7.

Intermediate contours generation among the sub-region was implemented using ACIS geometry modeler and Visual C++. The control points and knots extracted from sub-region contours were input to the program used to generate intermediate contours. A mid-contour between the consecutive sub-region contours is computed using straight-line homotopy method. The distance between each two consecutive contours is determined by the RP machine. The number of contours generated at this stage is equal to the number of layers in part build. The contour information can be as an input for the RP software.

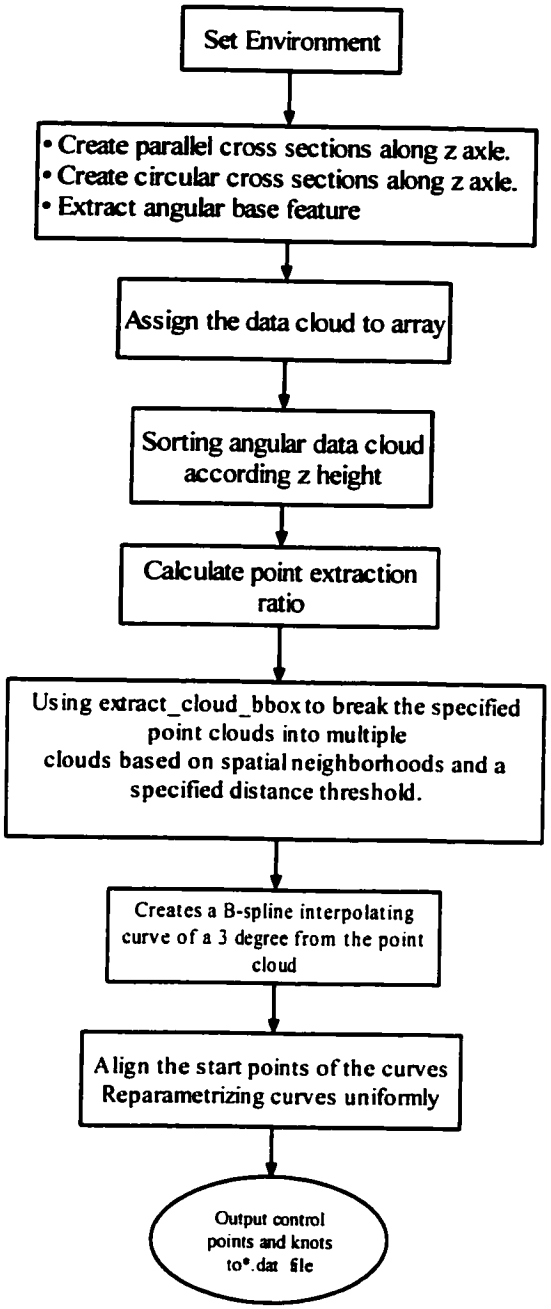


Figure 5.7. Scoll flow chart for direct contour generation

According to the contour information, the RP software will generate support contours, hatch contours, and output tool path to guide the machine movement.

Different parts are tested to demonstrate the accuracy and efficiency of this approach.

5.3.1 Example 1: A Nozzle

The nozzle was tested as the first example. Figure 5.8 (a) shows the initial scanned data cloud. In order to divide the data cloud into sub-section to generate sub-region, horizontal cross-section method is performed and the corresponding result is shown in Figure 5.8 (b) (20 cross-sections). Vertical cross-section and angular feature detection methods are used to calculate each horizontal cross-section's point extraction ratio. The sub-region contour data model is shown in Figure 5.8 (c).

5.3.2 Example 2: An Automotive Transmission Shift

In the second example, an automotive transmission shift is reverse-engineered in order to show the point reduction algorithm. The original data clouds (Imageware sample) of shift contain 65,770 points and occupy a volume of 56 × 72 × 121 mm as shown in Figure 6.9 (a). The point clouds are sliced horizontally into 25 cross-sections shown in Figure 6.9 (b). Curvature changing points are recognized by using the angular base feature method. The minimum angle is set at 5°. Using the extraction ratio of 0.3, sub-regions are created. Then each sub-region data cloud is interpolated to B-spline contour with 3 degree as shown in Figure 6.9 (c). In the end, the contour model is generated by applying straight-line homotopy method as shown in Figure 6.9 (d). Figure 6.10 illustrates the difference between a contour curve and corresponding cross-section data cloud. Auto-shift' surface model and STL mesh model are shown in Figure 6.11. Figure 6.12 shows the comparison of the maximum deviations of each contour (STL contour data and direct contour data) from associated cross section data cloud.

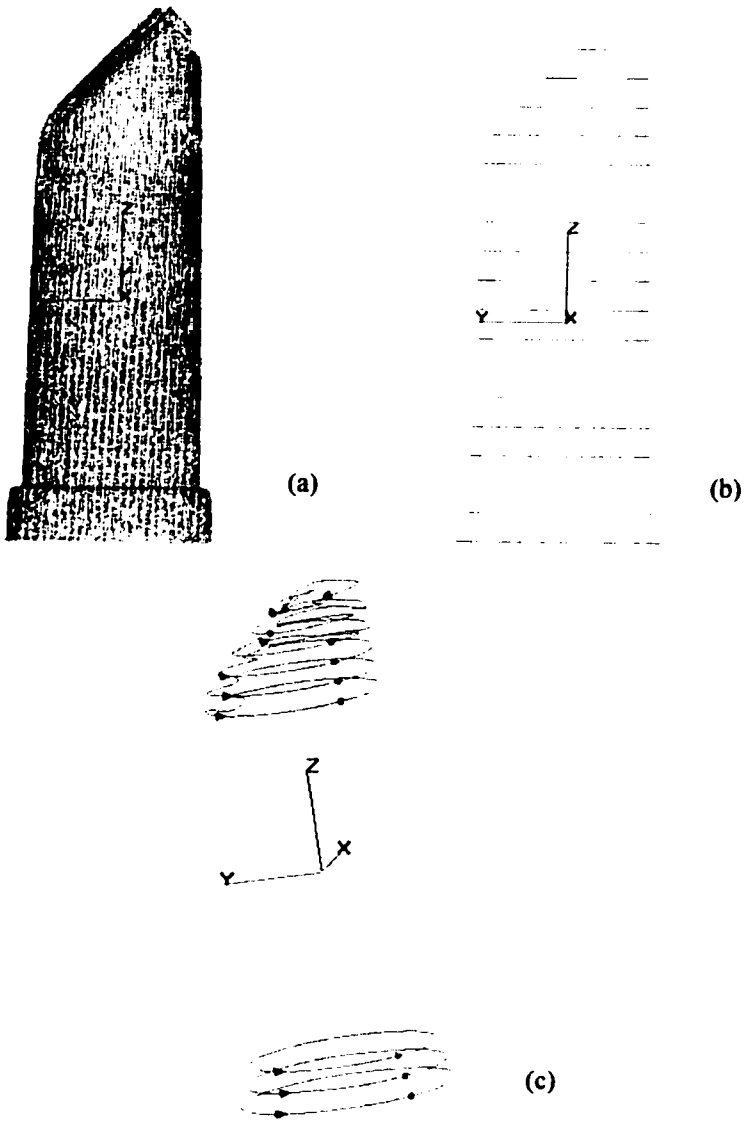


Figure 5.8. (a) Scanned data cloud of a nozzle (b) Horizontal cross-sections (c) Sub-region contours

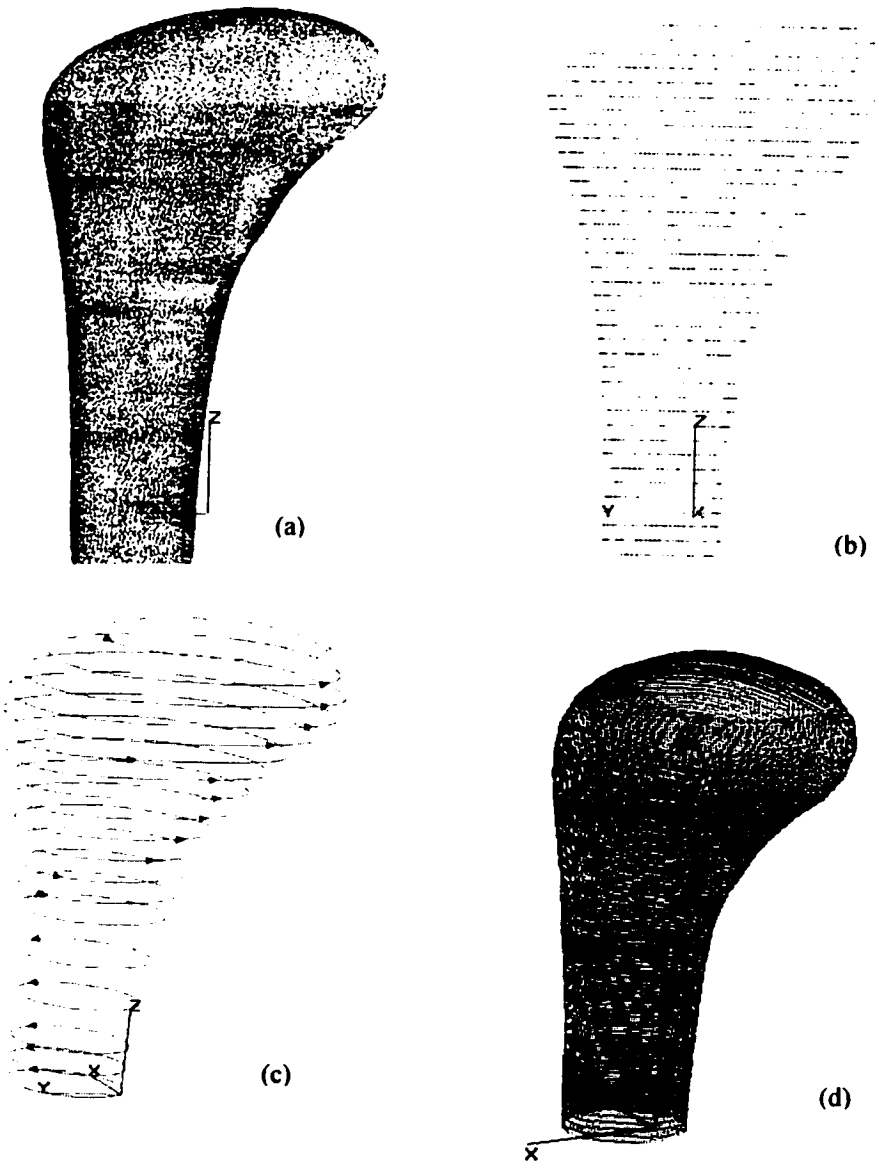
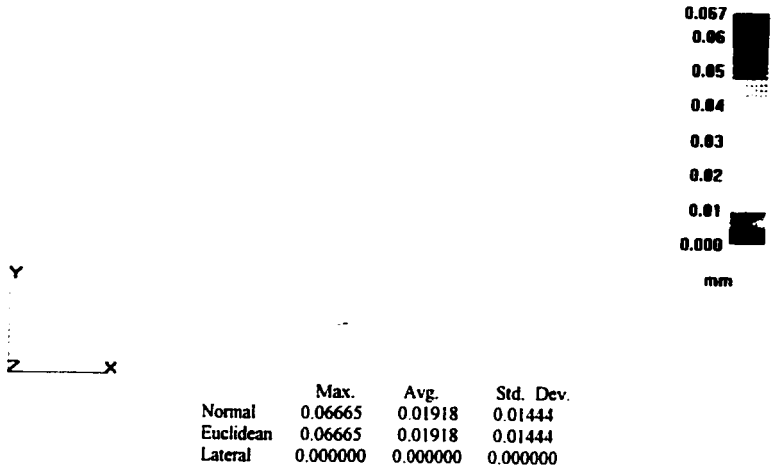


Figure 5.9. (a) Scanned data cloud (b) Horizontal cross-sections (c) Sub-region contours (d) Result of direct contour generation



	Max.	Avg.	Std. Dev.
Normal	0.06665	0.01918	0.01444
Euclidean	0.06665	0.01918	0.01444
Lateral	0.000000	0.000000	0.000000

Figure 5.10. Error analysis example

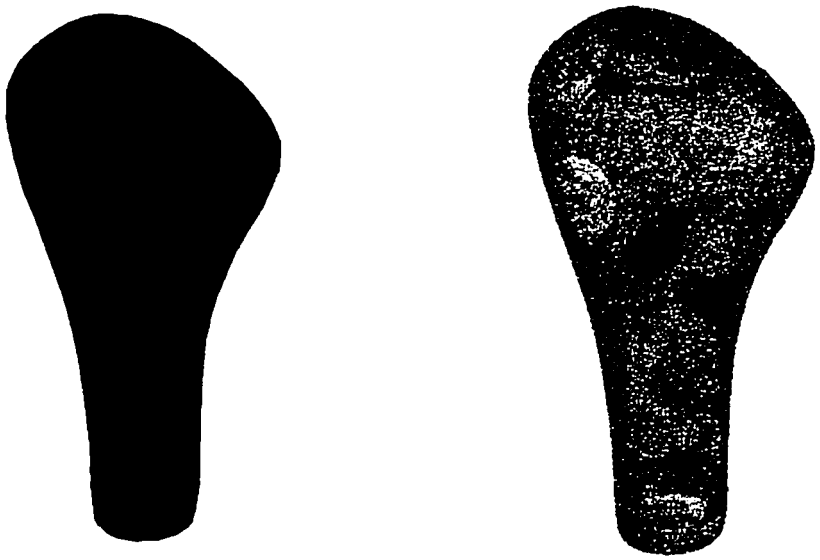


Figure 5.11. (a) Surface model of an auto shift (b) STL mesh model of an auto shift

Table 5.3. Deviation of STL model data and contour data model from cross-section data cloud

2.45	0.040422	0.01375	0.00588	0.003479	0.000801	0.000714
4.90	0.040422	0.01375	0.00588	0.004518	0.001047	0.000919
7.35	0.036719	0.013318	0.005883	0.003916	0.000913	0.000823
12.25	0.028629	0.011971	0.005278	0.002767	0.000789	0.000653
19.60	0.048039	0.012494	0.007214	0.004553	0.000955	0.000896
22.05	0.312164	0.108942	0.029778	0.004057	0.001086	0.000942
29.40	0.267072	0.022343	0.040093	0.00499	0.001056	0.000941
33.08	0.100929	0.015522	0.012482	0.009989	0.001484	0.00158
35.53	0.041218	0.016789	0.007753	0.007415	0.001792	0.001537
44.10	0.317732	0.036017	0.050134	0.017548	0.002324	0.003049
46.55	0.091837	0.062252	0.014518	0.043707	0.003443	0.005058

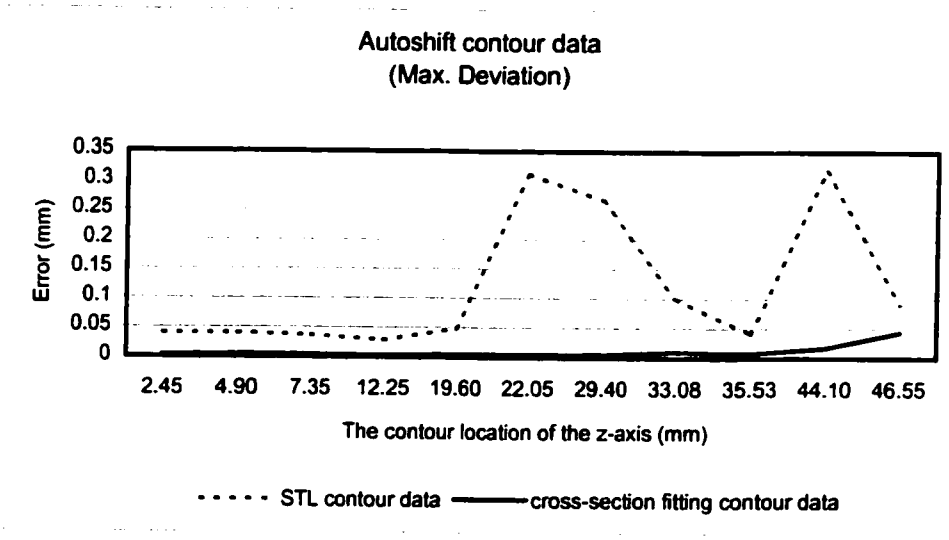


Figure 5.12. Comparison of maximum deviation between STL model and contour data model

5.4 DISCUSSION

This chapter presented implementation of the integration methods introduced in the last chapter. The methods were compared to the traditional method reported in the literature.

The first method, direct slicing, uses the exact representation of the slice contours which results in better accuracy of data for generating layers. The use of intersection routines within ACIS geometric modeler results in greater accuracy due to the stable representation of the model using B-spline. This approach eliminates the tessellation stage that is usually employed in traditional method. It also can be beneficial in terms of file size and processing time before the build can start. However, direct slice data are more difficult manipulate.

The second method presents a new approach for STL file construction directly from unconstructed data cloud scanned by the laser scanner. Using spatial filtering parameter, bin size, which is extracted from the data cloud, it removes the redundant points to speed up computation and reduce storage space. This approach differs from free-form surface modeling because it manipulates polygonal mesh data directly. For the user, it is intuitive and easy to control. It cuts out the need to construct CAD model, which needs special skills and considerable amount of time. Although polygon mesh generation frequently results in errors, like gaps, holes etc., it is easy to find commercial software to repair it.

The third method, direct contour generation, directly generate slice contour from data cloud. The slice contours are formed to three degree or higher degree curves. Straight-line homotopy method is used to generate intermediate contours. This method avoids the necessary to create a full 3D solid model and avoids the generation of the STL file and the possible need to correct errors after data transfer. The slice contours are represented by precise geometry representation without loss of precision inherent in STL. It also lessens the

time need to generate the STL file and repair the STL file. Like direct slice data, the output contour file is more difficult manipulate and the ability to manipulate the model and change build orientation is lost. The method is suitable for one-of-kind product, like bone, work of art etc.

CHAPTER 6

VISUAL SIMULATION FOR FDM

This chapter presents a visual simulation tool for FDM (Fused Deposition Modeling). The tool was developed to remedy the shortcoming of 'Catalyst' (Stratasys Inc.). The limitation of slicing algorithm in software sometimes results in error hatch vectors. 'Catalyst' can only show the tool path or STL model. The topological of each layer or final model to be built on a modeling system can't be displayed, hence, the error can not be detected.

This chapter discusses the implementation techniques involved in the development of the visual simulation tool. The visualization capability is demonstrated by comparing the virtual model with the physical model for several test cases.

6.1 OVERVIEW OF VIRTUAL PROTOTYPING

Visualization, in general, is a method of extracting meaningful information from complex data sets by using of interactive graphics and imaging. Virtual prototyping (VP) is the process of using a CAD model for testing and evaluation of specific characteristics of a product or a manufacturing process on the computer. The purpose of virtual prototyping is to reduce the iterations of design-prototype-test cycles. The further downstream conflicts occur,

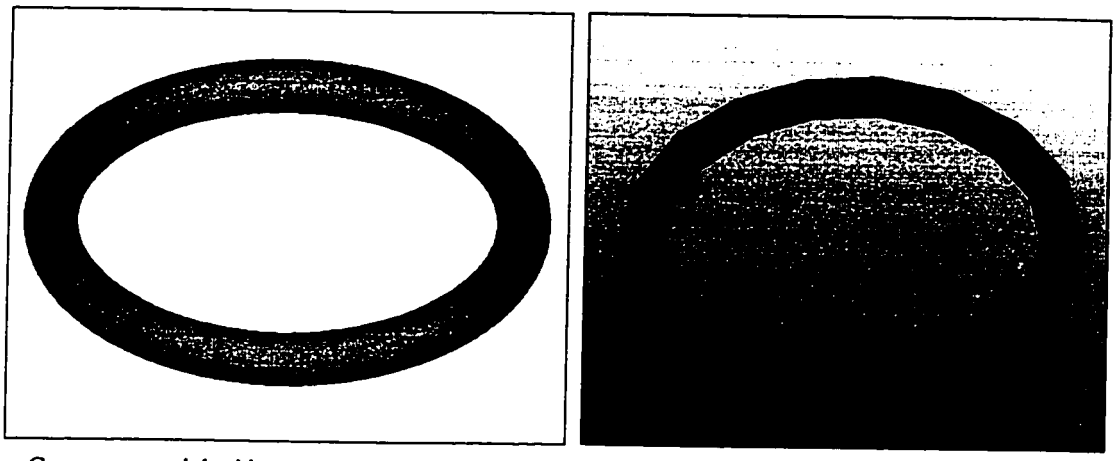
the more costly and time-consuming the iterations will be. Virtual prototyping in engineering can be classified into product design and process. In product design domain, VP is concerned with collision check, manufacturability assessments, ergonomic evaluation and functionality testing. The work in manufacturing process domain focused on visualization of the final product under specific operating conditions. Visual simulation tool for FDM belongs to later one. It may result in accurate determination of the process parameters, and hence reduce the number of costly physical prototyping iterations.

6.2 INTEGRATION OF VIRTUAL PROTOTYPING AND RAPID PROTOTYPING

The integration virtual prototyping and rapid prototyping allows a designer to assess and visualize the influence of process parameters on the part quality. It enable the design visualize the part before fabricating it.

Due to the advantages of virtual prototyping, some research group combined it with rapid prototyping at various stages. Traditional geometry-based modelers, which display a smooth, shaded object, can not provide information on the actual surface finish, like staircase effect, of the object shown as Figure 6.1. Chandru, et al. (1995) have proposed a voxel-base modeling for evaluating the geometrical effect on the surface of a physical part made by RP techniques. Jeem et al. (2000) proposed a visual simulation system for facilitating surface texture designs to be made by 3D printing. It is a voxel-base approach, which is only suitable for simple objects. Choim et al. (2001) developed a virtual prototyping test-bed for selection of optimal process parameters. The WorldToolKit (WTK) from 'sense8' is used to create the virtual world. The system can simulate SLA and SLS RP process. Qium et al. (1998)

developed an Intelligent Layered Manufacturing System for fabrication of multiphase electromechanical parts. A RP process virtual simulation package is also included in the system.



Geometry model without staircase effect Virtual prototyping model with staircase effect

Figure 6.1. Comparison of traditional geometry model and RP virtual prototyping model

The proposed tool in this work focuses on simulating the FDM RP process to remedy the shortcoming of 'Catalyst'.

6.3 METHODOLOGY

In FDM RP process, the nozzle through which material is extruded is attached to a liquefier head. The head moves so that the nozzle deposits the material along a particular path, which must be calculated in advance for each layer to be laid down. The path is called a hatch vector. The hatch vectors are obtained by passing rays onto the contour at grids of resolution of the hatch. A Ray traces the contour and the lines inside the contour are the hatch vectors. For each ray, the points of intersection with the part are stored. Two points define a

hatch vector. Hatch vectors are used to represent a volume (voxel) in 3D space. The height and width of the voxel are equal to the grid resolution as shown Figure 6.2.

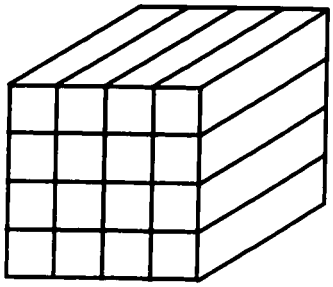


Figure 6.2. Voxel-base representation of volumes

Head trajectory may be represented by a hatch vector, which coincides with the center of the string. A slice can be considered a list of hatch vectors: (x_1, y_1) (x_2, y_2), ..., (x_i, y_i). A voxel can be built around each hatch vector with specific height and width. The voxel's width, height and length are the string width, the layer thickness and the length of hatch vector respectively, as shown by the voxel in Figure 6.3.

Building a voxel per hatch vector simulates the fused material solidification process. Thus a layer is formed by building voxel along the hatch vectors.

However, simulating a RP process involves processing a very large number of polygons. In order to maintain a satisfied frame-rate, the polygons should be reduced. It is obvious that voxel-to-voxel buildup is required only at the top of the layer which is being fabricated. The layers underneath are no longer required to represent a final part. Hence, the layers underneath can be simply represented by sweeping the contours by the thickness required.

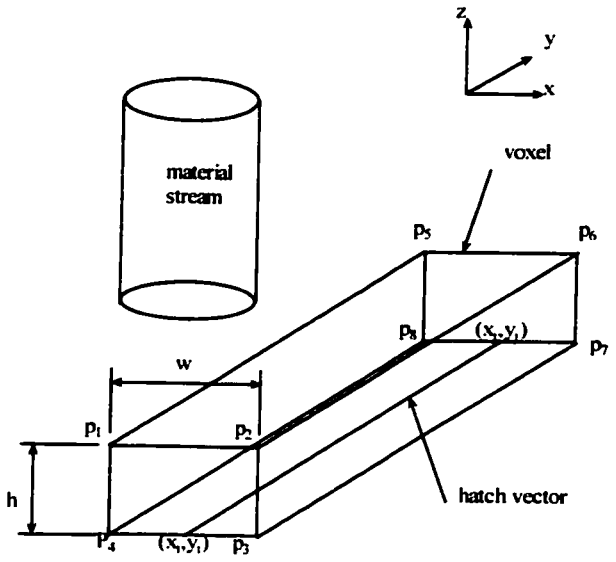


Figure 6.3. Head along a hatch vector

The hierarchical relationships of contours can get from the tool path file.

6.4 VISUAL SIMULATION OF RAPID PROCESS – FDM

Simulation process starts by importing a STL model into 'Catalyst'. In 'Catalyst', the STL model is oriented by a designer. After STL model's orientation is set, the software will automatically slice the STL model to create slice curves. The slice curves need to be non-overlapping and non-intersecting. Even though 'Catalyst' slicing procedures can handle gaps, overlaps, and other flaws to produce the desired closed curves, Checking and repairing the STL model before loading to Catalyst is recommended. Support generation is vital to ensure part integrity during the part build. The 'Catalyst' can generate supports that the model needs to ensure a quality build. Toolpath generation is the final stage in Catalyst. At this stage, the model has been process into a set of thick interior layers or set of thinner shell layers. The

interior layers are filled using a coarse, sparse pattern and the exterior layers are filled using a fine, dense pattern as shown in Figure 6.4.

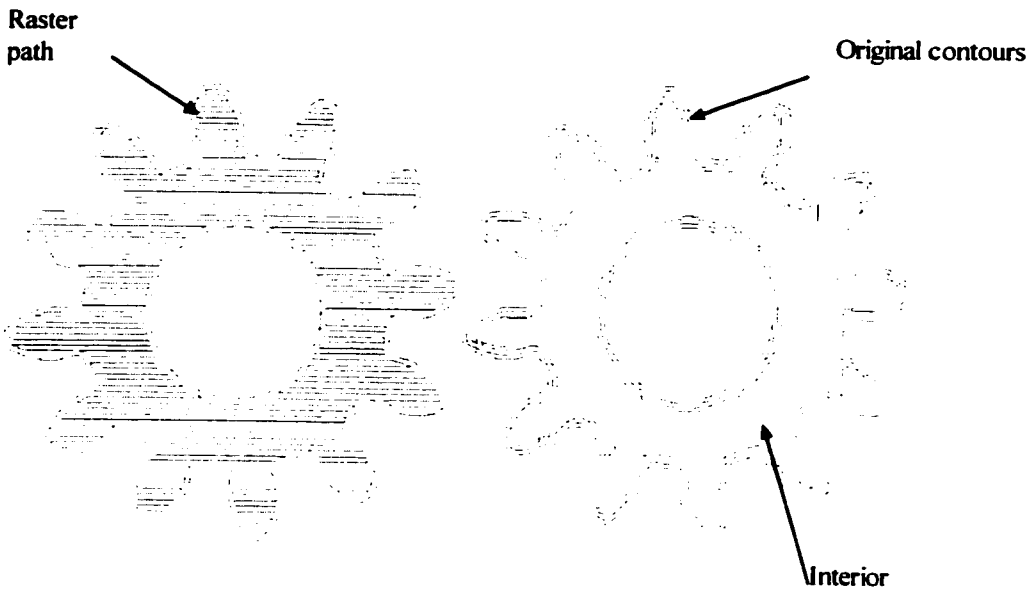


Figure 6.4. Toolpath example with method exterior/interior

The output of 'Catalyst' is toolpath file, CMB file. CMB file is a compressed machine code guiding machine head movement to build final parts. In order to extract information from CMB file, a file conversion model is used to transfer CMB file to readable ASCII *.dat file as shown in Figure 6.5.

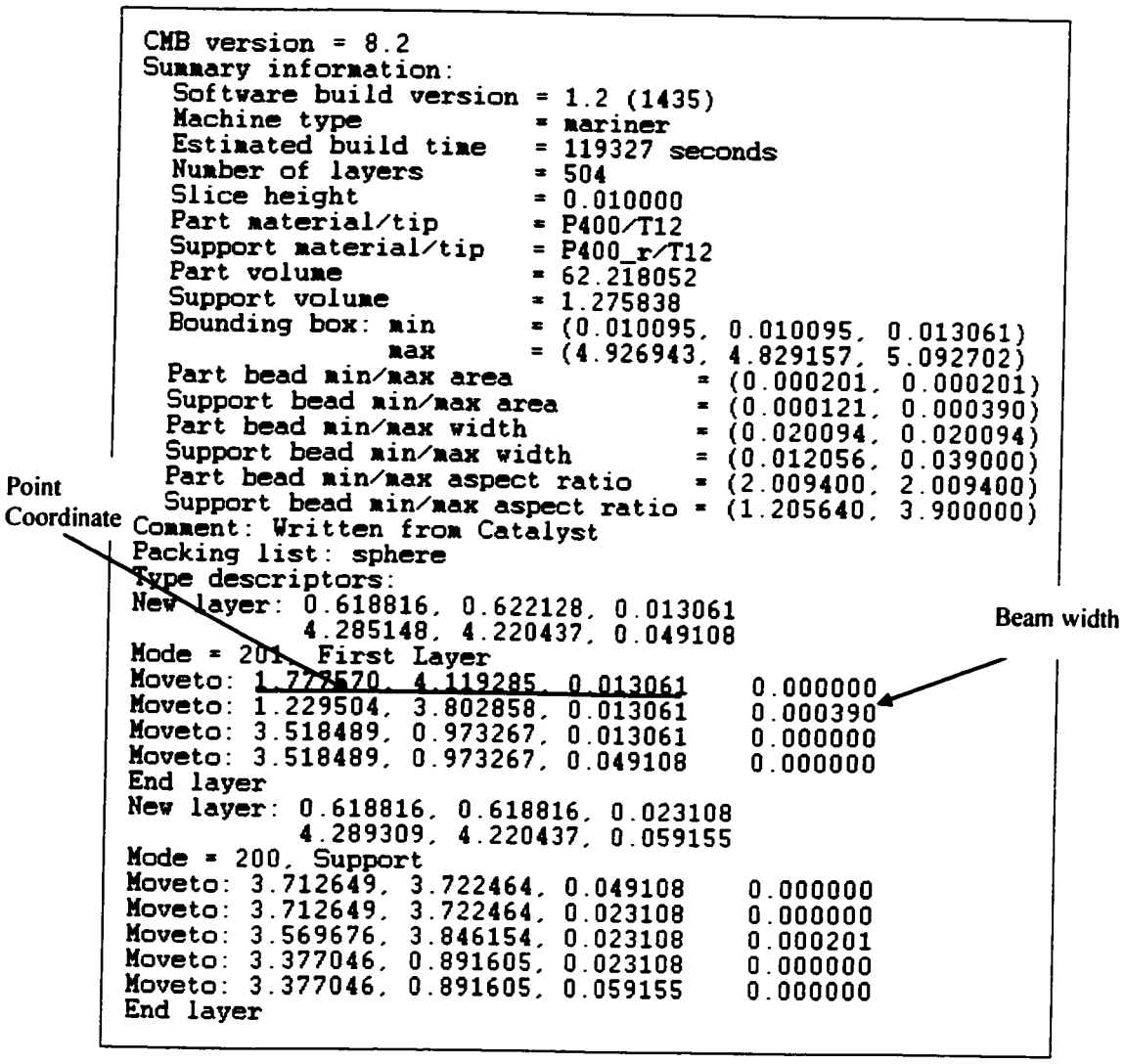


Figure 6.5. *.dat file sample transferred from CMB file

Reading data file, each two continued points are regarded as a hatch vector. Each hatch vector is used as a base to build voxel. The process is repeated until all the hatch vectors are converted into voxel in one layer. Once a layer is generated, the contours are swept to generate a layer. The hierarchical relationship of contours is to facilitate sweeping of a layer to determine which side of triangles should be rendered. The topological relationship among contours can extract from data files.

The whole simulation process is shown in Figure 6.6

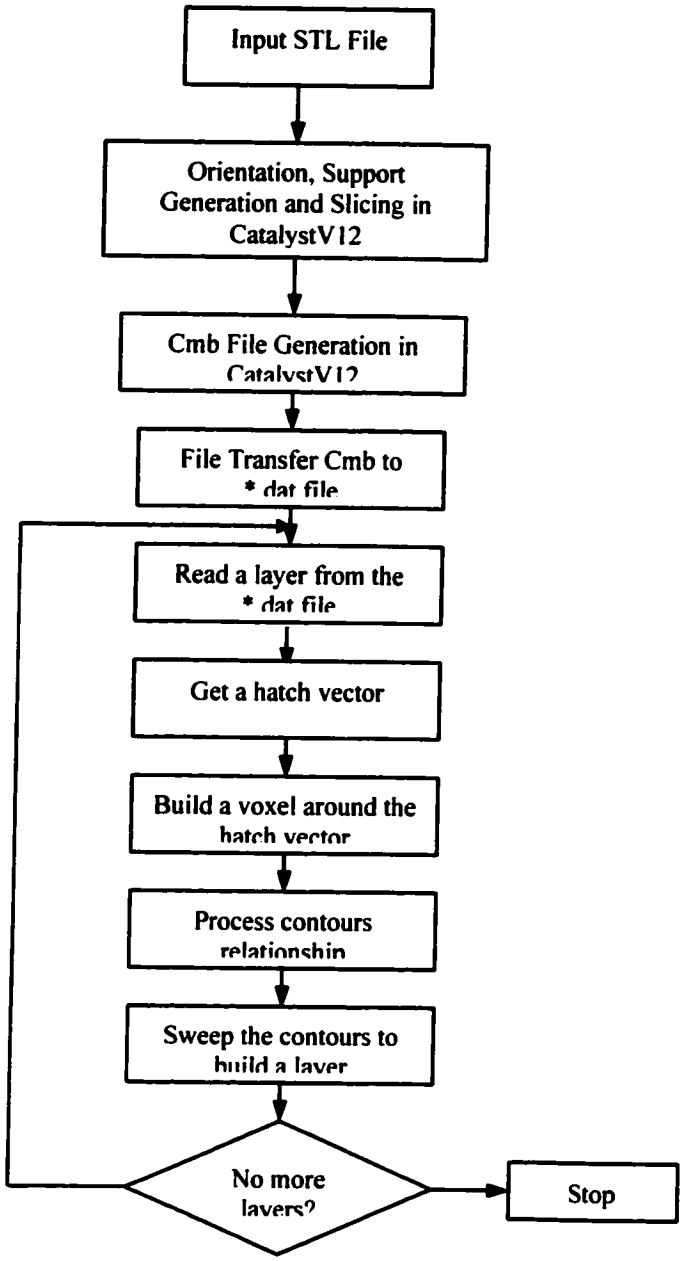


Figure 6.6. Flow chart for the virtual simulating FDM process

6.5 IMPLEMENTATION

Two methods were used to develop RP virtual prototyping tool. First method selects ACIS 3D toolkit and Virtual C++ as development tools. ACIS is an object-oriented three-

dimensional (3D) geometric modeling engine from *Spatial*. It is designed for use as the geometry foundation within virtually any end user 3D modeling application Figure 6.7 illustrates the overall process. This method can give excellent RP virtual prototyping model. However, using CAD modeling method to generate virtual prototype is only feasible to simulate relatively simple models. Generally, in order to represent parts accurately, the data sets of CAD models are very dense. Processing extremely dense data sets needs intensive computation. We tried to use this method to process more complicated model on a PC with 128 RAM and suffered insufficient memory. The other defect of this method is that it only simulates the final model. The part build process cannot be visual simulated.

The second method was developed using OpenGL and Visual C++ 6.0 on a desktop PC. The operating system of PC is Windows 98.

OpenGL (for "Open Graphics Library") is a software interface to graphics hardware. The interface consists of a set of hundreds procedures and functions to allow a programmer to specify the objects and operations involving in producing high quality graphical image, specially color images of three-dimensional objects. OpenGL includes a broad set of rendering, texture mapping, special effects, and other powerful visualization functions.

In this work, three OpenGL libraries (OpenGL32.lib, GLu32.lib and GLaux.lib) were linked to Visual C++ to set up an OpenGL window environment.

The visual simulation tool provides necessary graphical tools: coloring & shading, solid modeling, 3D viewing, and animation tools. The simulation opens a graphical window, specifies a color set for material, and starts the "material deposition" process

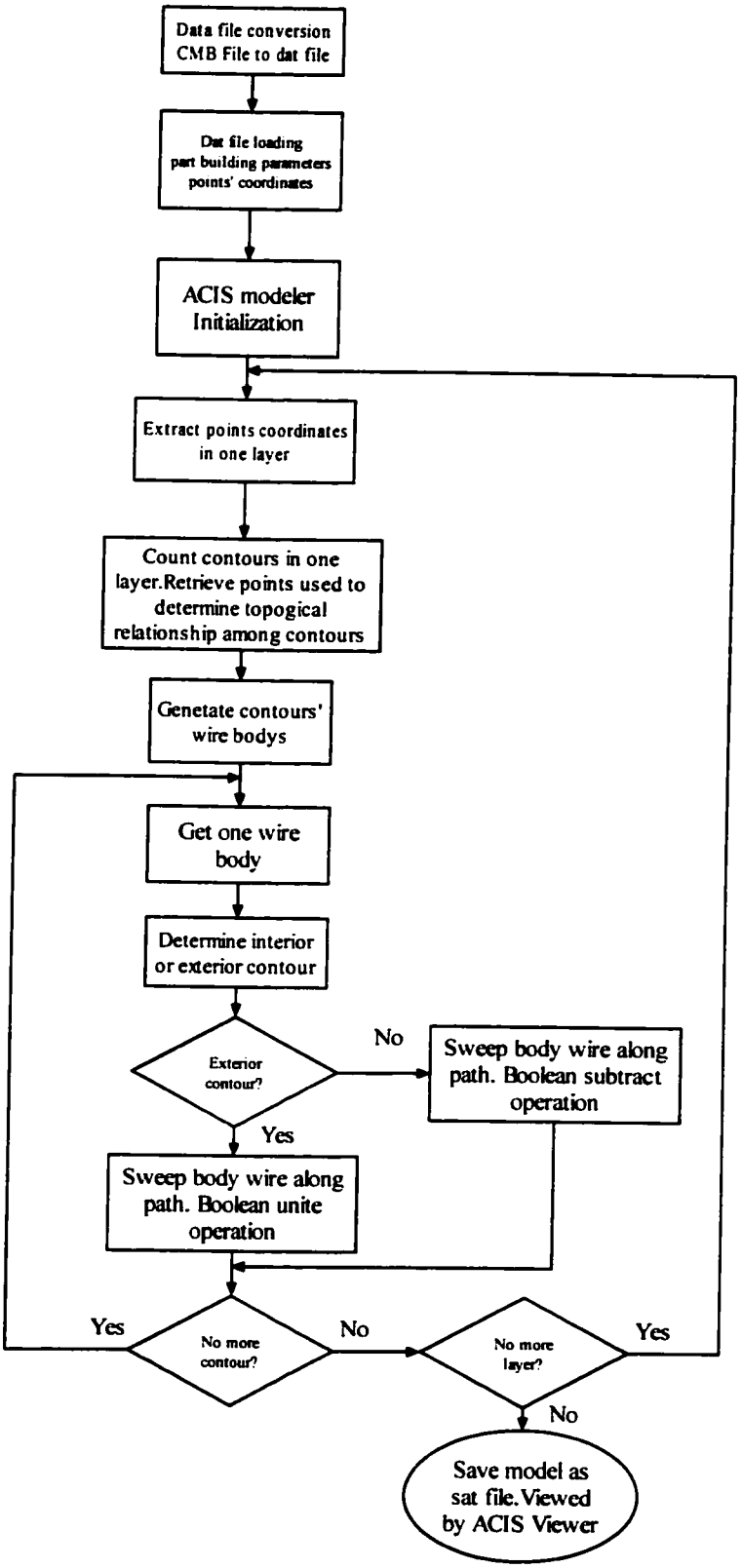


Figure 6.7. Flow chart for the RP virtual prototyping using ACIS 3D toolkit

along the toolpath on the starting layer. When one layer is finished, a new layer will be "deposited" on top of the finished layer, until all the layers of the part are stacked sequentially.

6.5.1 Example 1: Gear

6.8 shows a virtual fabricate gear. The virtual prototyping tool can show the whole part build process. Once the simulation is completed, the operator can rotate the part to view it or can map surface texture to get a realistic perception of the physical part, which the RP machine will build.

6.5.2 Example 2: Engine Block Sand-Casting Pattern Plates

A new approach has been devised for building pattern plates in a sand-casting operation by means of a rapid prototyping process (FDM). The advantage to this approach is the shot time needed to produce the pattern plate compared to that in conventional manufacturing.

Two surface models were selected to fabricate pattern plates as shown in figure 6.9. The surface models were saved as IGES files. Because Catalyst only accepts STL models, a file conversation process is needed to transfer IGES file to STL file before using Catalyst. Most commercial CAD systems can transfer IGES files to STL files, however, due to weak tessellation algorithm in the CAD systems, bad .STL files, which

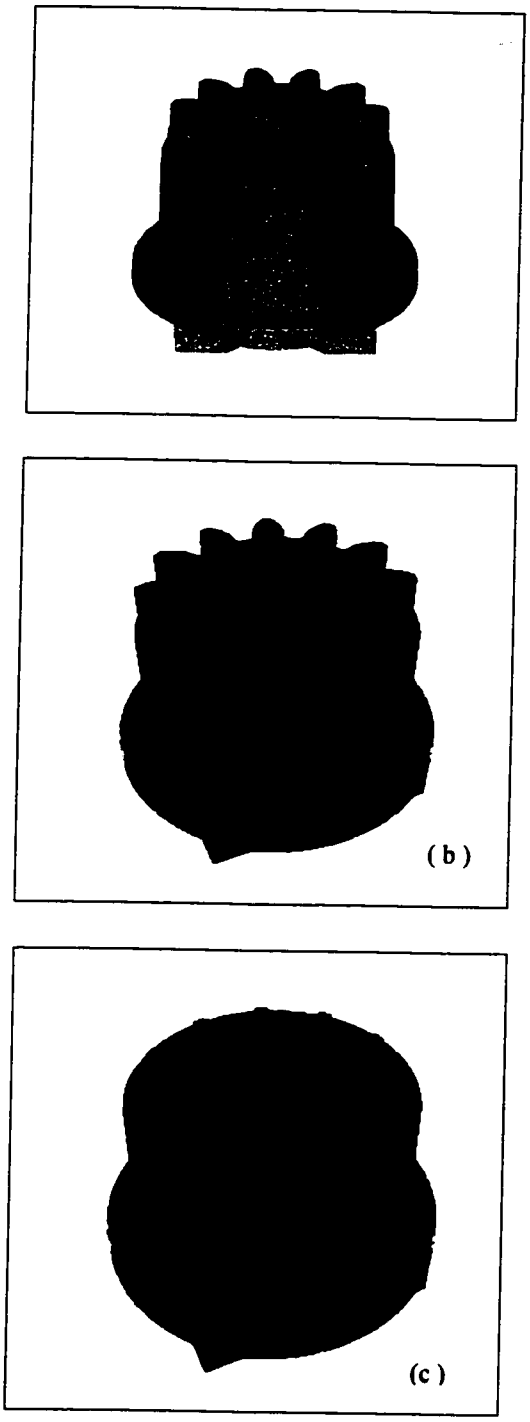


Figure 6.8. (a) STL model of a gear. (b) A virtually fabricated
(c) Build process simulation

contain flipped normals facets and non-closed shells (presence of gaps or holes), may be generated during file conversation. '3Data Expert' (DeskArtes) was selected in this work for file conversation. DeskArts '3Data Expert' is a tool for repair, conversion and manipulation of 3D CAD file. IGES conversion to STL and repair are available for the 3Data Expert.

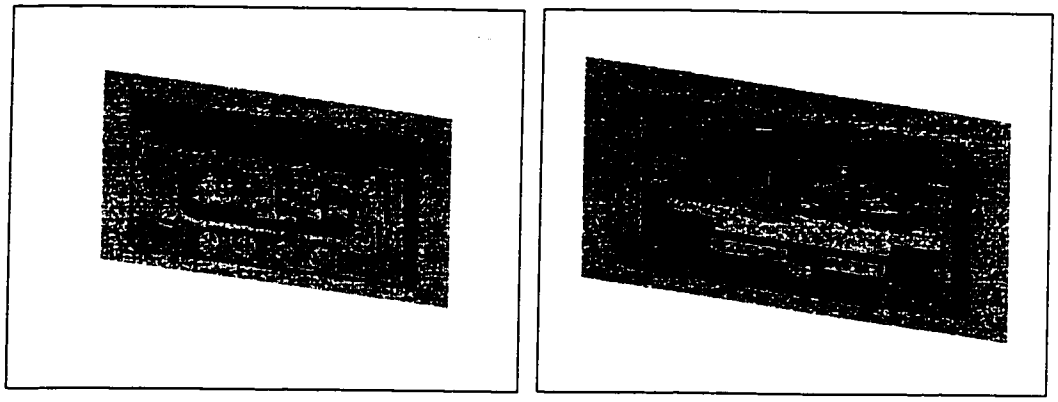


Figure 6.9. Engine block pattern plates' surface models

Once file conversation and STL verification are completed, we used Catalyst to preprocess STL files and generate toolpath files (CMB). CMB viewer, one tool in 'Catalyst' package, was used to check toolpath files as shown in Figure 6.10.

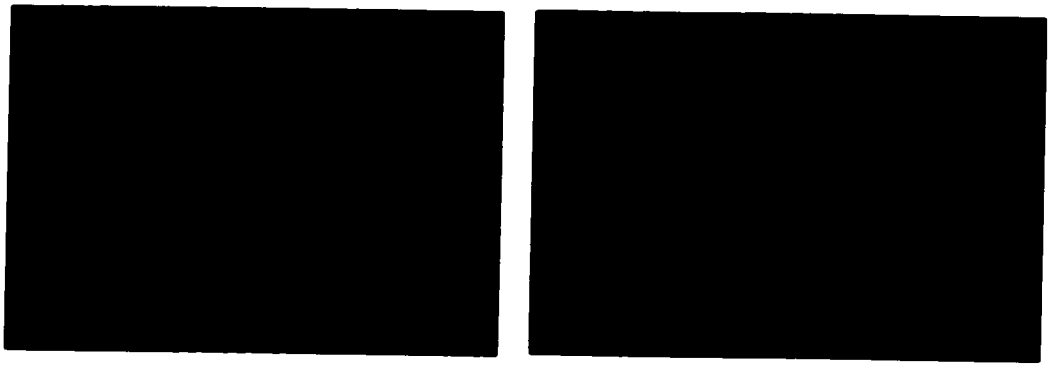
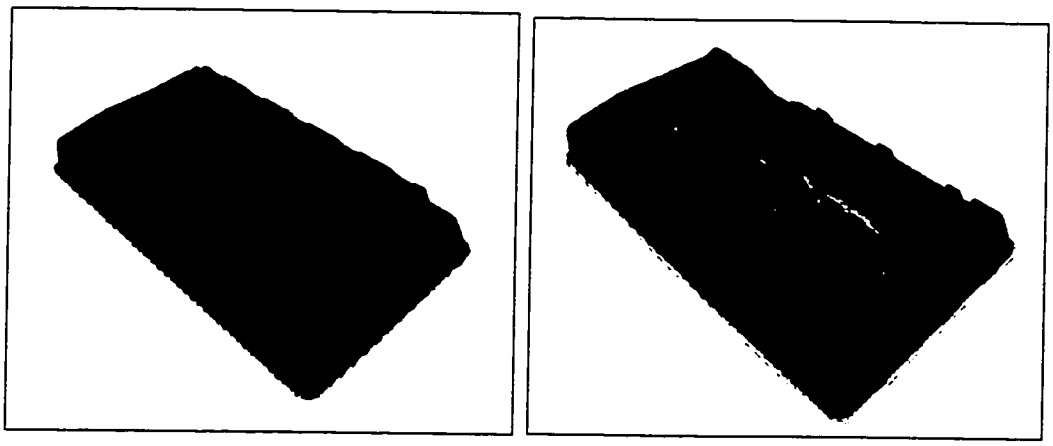


Figure 6.10. Slice contours of pattern plates

CMB viewer only displays polyline contours. A realistic perception of the physical part, which the RP machine will build, cannot be generated. The virtual prototyping tool remedies the shortcoming of CMB viewer. Figure 6.11 shows the virtual prototyping models by using the virtual prototyping tool.



(a) (b)

Figure 6.11. Virtual prototyping models of pattern plates before correcting

Checking the VP models, we can find that some cavities in original surface models are filled by build material. Part b is separated into two pieces. The reason resulted in this failure is incorrect processing contour topology. Catalyst uses odd-winding rule described in

Chapter 4 to determine the hierarchical relationship of contours. In watertight surface, odd-winding rule is always feasible. But in this case, because the surface models are not closed, using odd-winding rule in following situation will result in building incorrect parts.

In order to avoid this error, two methods were selected to try to correct the surface models. The first method is by offsetting the surface. The purpose of surface offset is to generate thickness on surface. Then an offset surface can be seen as a thin walled surface. However, one problem in offset surface generation arises when the distance between the offset and surface is less than the given offset distance in some section of the surface. This creates self-intersections in the offset surface. Alexander and Dutta (2000) presented a local wall thickening algorithm to preprocess sheet surfaces. But the algorithm can only handle simple features. Hence, offset surface is unfeasible in this case.

Another method is adding surfaces along the part profile. Figure 6.12 shows the corrected surface models using second method. Using this method, we successfully build the parts we need. Figure 6.13 illustrates virtual prototyping models to be built by 'Prodigy'.

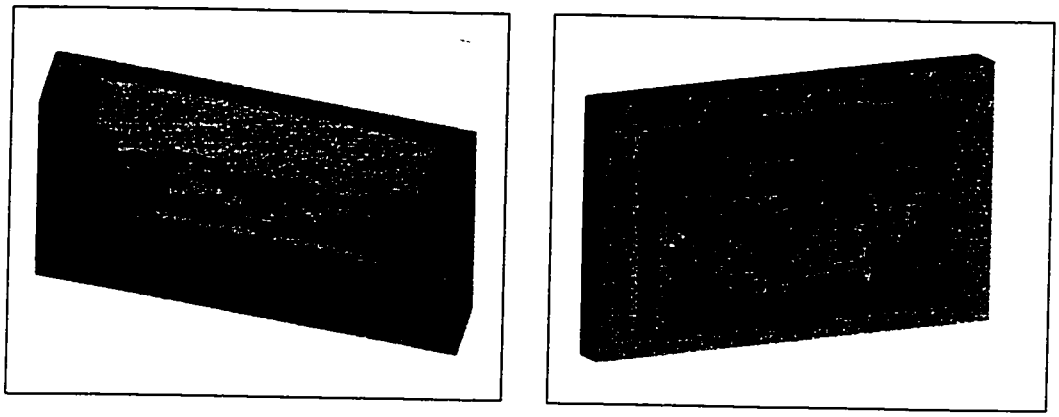


Figure 6.12. Surface models of pattern plates after correcting

6.6 DISCUSSION

The virtual prototyping tool facilities the visualization of virtual fabricated parts. The tool developed in this work simulates an RP (FDM) process with actual physical phenomena. This provides a designer a tool to visualize and verify the unseen fabrication capabilities of an RP machine. Virtual fabrication is simulated using voxels approach. The techniques are able to hide the level-of-details efficiently and represent the final part accurately. This tool can also be used to produce color prototypes and multi-material prototypes. Color prototypes help the designer visualize the parts more effectively. Multi-material prototyping is an emerging technique. Using multi-material objects, functionally efficient and cost-reducing designs can be realized.

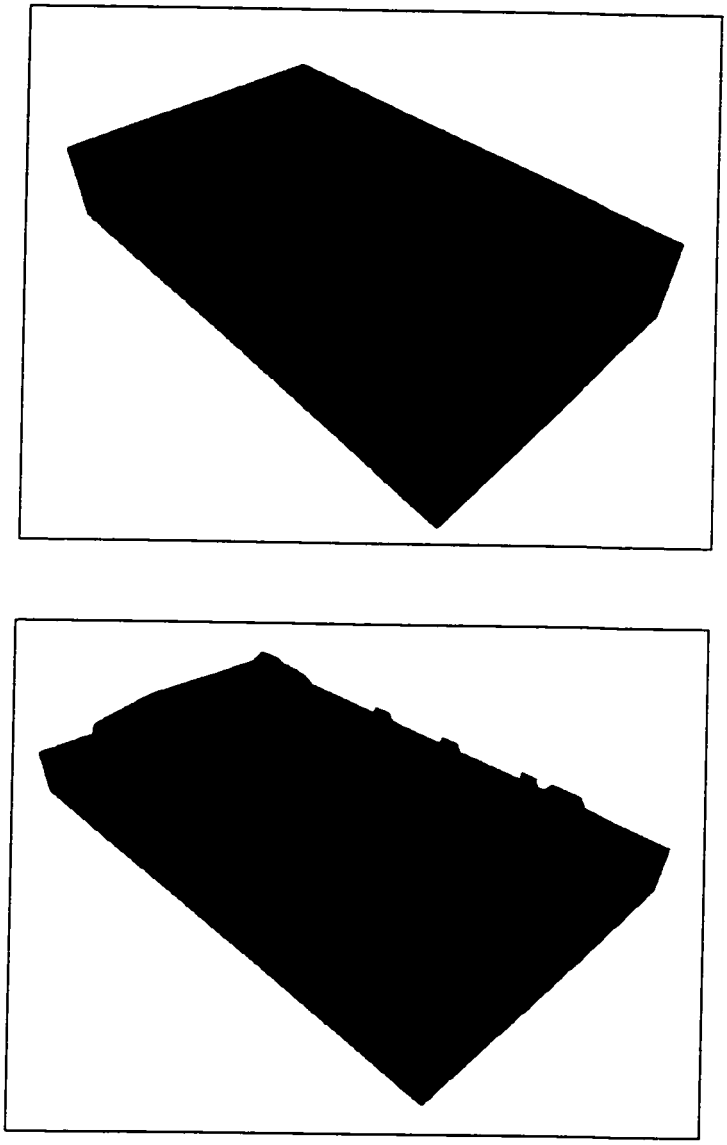


Figure 6.13. Virtual prototyping models of pattern plates after correcting

CHAPTER 7

CONCLUSION

This chapter comprises three sections: contributions, conclusions and future research.

7.1 CONTRIBUTIONS

This thesis has made the following contributions:

1. Three methodologies for the integration reverse engineering and rapid prototyping were modified and implemented.
   1) An approach using data cloud to generate 3-D model then direct slicing.
   2) An approach using data cloud to generate STL file then slicing.
   3) An approach using data cloud to directly generate slice contour file.
2. A visual simulation tool for simulating FDM (Fused Deposition Modeling) process was developed and implemented. This tool can be a compensatory tool for 'Catalyst' (A software used to preprocess STL models and generate toolpath files to be built on a *Stratasys* 'Prodigy' - 3D printer).

7.2 CONCLUSIONS

There is a need for direct slicing CAD models. The first method, direct slicing, can be beneficial in terms of file size and in cutting out the need for data transfer from CAD models to tessellated models. Accuracy can be enhanced, especially on rounded or tubular designs. NURBS (Non-Uniform Rational B-Splines) representing contours result in more accuracy for generating layers. However, output file format selection is a significant problem with this approach. Most of contour file formats only have the capability of producing polylines of the contour of the slice. By converting the curve to segments of straight lines, accuracy, a major advantage over the STL file format is lost. But, the confidence in terms of not exceeding the expected error is greater with the combination of direct slicing and polyline represented file format. Because the process of slicing contour generation is the process of calculating the intersection between plane and B-spline surface, the computation load is higher than triangular mesh slicing, which calculates the intersection between line to line.

The second approach for achieving surface construction from unconstructed 3D cloud, with accuracy control, has been developed. The method starts with analyzing the deviation between triangular mesh and the underlying surface. The triangular facet edge length is determined by surface curvature and the allowed tolerance. A data reduction process is employed using voxel bin size decided by triangular facet edge length. A triangulation algorithm is applied to the reduced data cloud to generate triangular mesh. Triangular mesh can be directly output as the STL format to bridge downstream process and rapid prototyping. This method is well behaved when applied to data clouds collected from surfaces, such as the light cover and the face mask, with relatively low curvature and relatively simple topology. Surfaces with holes, sharp edge, very high curvature can not be

handled using this method. This method simplifies the problem of constructing CAD model from scanning data cloud. The surface remains as triangular mesh data from start to finish. The user can balance the tradeoff between model resolution and file size using refinement and decimation tools. Even for non-closed surface mesh, it is possible to calculate model volume by offsetting a wall thickness and then outputing a STL file. With STL file, user can easily fabricate physical parts using one of RP systems.

The direct contour generation method, as the third method, uses curvature information extracted from data cloud to achieve data reduction. Straight-line homotopy is used to generate intermediate contours. This method has been applied to two example parts. The results show that contour models have smaller storage requirement and better accuracy. It is especially suitable for rounded or tubular parts. This method avoids creation of a full 3D solid model, which takes considerable amount of time and needs special skills, or generation of STL file and the possible need to correct errors in data transformation. Slice data are represented by precise geometry representation (NURB-base B-rep) without loss of precision inherent in STL model. It also may lessen the time needed to generate the STL file. The primary disadvantage of this method is that the ability to manipulate the model and change build orientation is lost. Some technologies need to be based on a CAD model data approach rather than slicing data.

The proposed FDM process visual simulation technique provides the part designer with a valuable tool for interactions between virtual and real world and helps realize a manufacturable design with minimum iteration by taking the full fabrication process rules of FDM into account. With the implemented visual simulation tool, the designer can figure out an approximated visual image to the physical model before actually fabricating it. It

incorporates parameters of the FDM process and hence provides designers with a valuable tool for verifying unseen fabrication capability from FDM build file.

7.3 RECOMMENDATIONS FOR FUTURE WORK

This thesis presents a broad view of two emerging techniques: reverse engineering and rapid prototyping. A number of issues, which provide future research related to this work, have been identified.

1. Enhancing triangulation algorithms to deal with surfaces with holes, sharp edge, very high curvature.

2. Enhancing direct contour generation algorithm to deal with surfaces with complex topology (multiple contours in each cross-section), and holes.

3. Packaging the developed algorithms and integrating them with Surfacer to construct a Integrated RE & RP System, which provides alternate approaches for integrating reverse engineering and rapid prototyping.

4. Packaging the virtual simulation tool and embedding it in Catalyst.

REFERENCES

Alciatore, D.G., Wohlers, T.T. (1996) 'Importing and reshaping digitized data for use in rapid prototyping: a system for sculpting polygonal mesh surfaces', *Rapid prototyping journal*, Vol 2, No.1, pp.13-23.

Aluru, R., Keefe, M. and Advani, S. (2001) 'Simulation of injection modeling into rapid prototyped model' *Rapid Prototyping Journal*, Vol.7 No.1, pp.42-51.

Barequet, G., Sharir, M. (1995) 'Filling gaps in the boundary of a polyhedron', *Computer aided geometric design*, 12, 207-229.

Bhashyam, S., Shin, K.H. and Dutta, D. (2000) 'An integrated CAD system for design of heterogeneous objects', Rapid prototyping journal, Vol6, No.2, pp119-135.

Bohn, J.H., Wozny, M.J. (1992) 'Automatic CAD-model Repair: Shell-Closure' *Proceedings Solid Freeform Fabrication Symposium*, pages 86–93. University of Texas at Austin, August 1992.

Bradley, C. (2001) 'Rapid prototyping models generated from machine vision data', *Computers in Industry*, 44(2), 2001, pp. 159-173.

Chandru, V., Manohar, S., Parkash, C.E. (1995) 'Voxel-based modeling for layer manufacturing' *IEEE Computer Graphics and Applications* 11, pp.42-47.

Chen, L.C., Lin, C.I. (1997) 'An integrated reverse engineering approach to reconstructing free-form surfaces', *Computer Integrated Manufacturing Systems*, Vol.10, No.1, pp.49-60.

Chen, L.C., Lin, C.I. (2000) 'Reverse engineering in the design of turbine blades – a case study in applying the MAMDP', *Robotics and Computer Integrated Manufacturing* 16, pp.161-167.

Chen, Y.H., Ng, C.T. (1997) 'Integrated reverse engineering and rapid prototyping', *Computer Ind. Engng* Vol. 33 No.3-4, pp.481-484.

Chen, Y.H., Wang, Y.Z. (1999) 'Genetic algorithms for optimized re-triangulation in the context of reverse engineering', *Computer-Aided Design*, 31(4), pp. 261-271.

Chen, Y.H., Song, Y. (2001) 'The development of a layer based machining system', *Computer-Aided Design*, 33 (4), pp. 331-342.

Chivate, P.N., Jablokow, A.G. (1995) 'Review of surface representation and fitting for reverse engineering', *Computer Integrated Manufacturing Systems*, Vol.8, No.3, pp.193-204.

Chiu, W.K., Tan, S.T. (1998) 'Using dexels to make hollow models for rapid prototyping', *Computer-Aided Design*, 30 (7), pp. 539-547.

Chiu, W.K., Tan, S.T. (2000) 'Multiple material objects: from CAD representation to data format for rapid prototyping', *Computer-aided Design*, 32 pp.707-717.

DeskArtes (2001) 'DeslArtes expert series 4.3 reference manual', www.deskartes.com.

Dolenc, A., Mäkelä, I. (1994) 'Slicing procedure for layered manufacturing techniques', *Computer-Aided Design*, Vol. 26, No. 2 pp. 119-126.

Drakos, N. (1994) 'Computer Based Learning Unit', LaTeX2HTML Usage Statistics. Internal Report, University of Leeds.

Dutta, D., Kumar, V., Pratt, M.J. and Sriram, R.D. (1998) 'Towards STEP-based data transfer in layered manufacturing' *Proceedings of the 10th international IFIP WG5.2/5.3 conference*, Prolamat.

Fadel, G.M., Kirschman, C. (1996) 'Accuracy issues in CAD to RP translations' *Rapid Prototyping Journal*, Vol.2, No.2 pp. 4-17.

Imageware (1998) 'Imageware Surfacer version 8.0 tutotial' Imageware, a Division of SDRC (Structural Dynamics Resarch Corporation).

Jacob, G.K., Kai, C.C., Mei, T. (1999) 'Development of a new rapid prototyping interface', *Computer in industry* 39 pp.61-70.

Jamieson, R., Hacker, H. (1995) 'Diract slicing of CAD models for rapid prototyping', *Rapid prototyping journal*, Vol 1, No.2 pp. 4-12.

Jee, H.J., Sachs, E. (2000) 'A visual simulation technique for 3D printing' *Advance in Engineering Software* 31 pp.97-106.

Jurrens, K.K. (1999) 'Standards for the rapid prototyping industry', *Rapid prototyping journal*, Vol 5, No.4 pp.169-178.

Kemmerer, S.J. (1999) 'STEP: The Grand Experience', *NIST Special Publication SP 939*, US Government Printing Office, Washington, DC 20402, USA.

Kirschman, C.F., Jara-Almonte, C.C. (1992) 'A parallel slicing algorithm for solid freeform fabrication processes', *Solid freeform fabrication symposium 1992*, University of Texas, Austin, pp.1-7.

Koc, B., Ma, Y., Lee, Y.S. (2000) 'Smoothing STL files by Max-Fit biarc curves for rapid prototyping'. *Rapid prototyping journal*, Vol 6, No.3 pp.186-203.

Kochan, D. (1992) 'Improved quality of SMF-procedures by systemized operational planning', *Solid freeform fabrication symposium 1992*, H.L. Marcus et al., eds., University of Texas, Austin, pp.34-43.

Kruth, J.P., Kerstens, A. (1998) 'Reverse engineering modelling of free-form surfaces from point clouds subject to boundary conditions', *Journal of Materials Processing Technology* 76 pp.120-127.

Kulkarni, P., Dutta, D., (1996) 'An accurate slicing procedure for layered manufacturing', *Computer-aided Design*, Vol.28, No.9 pp.683-697.

Kulkarni, P., Marsan, A., Dutta, D., (2000) 'A review of process planning techniques in layered manufacturing', *Rapid Prototyping Journal*, Vol.6 No.1 pp.18-35

Kumar, V., Dutta, D. (1997) 'An approach to modeling multiple material objects', Proceedings of the 4th ACM Solid Modeling Symposium Atlanta, pp. 336–45.

Laszlo, M.J. (1996) 'Computational geometry and computer graphics in C++', Upper Saddle River, N.J. Prentice Hall.

Lee, K.H., Woo, H. (1998) 'Use of reverse engineering method for rapid product development'. *Computer ind. Engng* Vol. 35 Nos 1-2, pp.21-24.

Lee, K.H., Woo, H. (2000) 'Direct integration of reverse engineering and rapid prototyping', *Computer & industrial engineering* 38 pp.21-38.

Lin, C.Y., Liou, C.S., Lai, J.Y. (1997) 'A surface-lofting approach for smooth-surface reconstruction from 3D measurement data', *Computers in Industry* 34 pp.73-85.

Ma, W.Y., He, P. (1999) 'An adaptive slicing and selective hatching strategy for layered manufacturing', *Journal of materials processing Technology*, 89-90 pp. 191-197.

Mäkelä, I., Dolenc, A. (1993) 'Some Efficient Procedures for Correcting Triangulated Models', *Solid Freeform Fabrication Symposium 1993*, H. L. Marcus et al., eds. University of Texas, Austin, pp. 126-134.

Mani, K., Dutta, D. (1999) 'Region-based adaptive slicing', *Computer-aided Design*, Vol.31, pp317-333.

Marsan, A.L., Kumar, V., Dutta, D., Pratt, M.J. (1998) 'An assessment of data requirements and data transfer formats for layered manufacturing', NISTIR 6216.

Microsoft (1998) 'MSDN library visual studio 6.0', Microsoft Corporation.

Mortenson, M.E. (1985) 'Geometric modeling', John Wiley & Sons, Inc.

Morvan, S.M., Fadel, G.M. (1996) "IVECS, Interactively Correcting STL Files in a Virtual Environment," *Solid Freeform Fabrication Symposium 1996*, D. L. Bourell et al., eds. University of Texas, Austin, pp. 491-498.

Motavalli, S. (1998) 'Review reverse engineering approaches', *Computer ind. Engng* Vol. 35 Nos 1-2, pp.25-28.

Owen, J.C. (1994) 'Interactive feature-based reverse engineering of mechanical parts', Proceedings of the Image Understanding Work-shop, mechanical parts, CA, pp.1115–1124.

Pham, D.T., Gault, R.S. (1998) 'A comparison of rapid prototyping technologies', *International journal of machine tools & manufacture* 38 pp.1257-1287.

Piegl, L.A., Richard, A.M. (1995) 'Tessellating trimmed NURBS surfaces', *Computer-Aided design* Vol.27 No. 1 pp16-26.

Qiu, D., Langrana, N., Danforth, S., Jafari, M., Safari, A. (1998) 'Virtual Simulation for Multi-material LM Process', *Proceedings of the Solid Freeform Fabrication Symposium*, August 1998.

Rajagopalan, M., Aziz, N.M., Huey, C.O. (1995) 'A model for interfacing geometric modeling data with rapid prototyping systems', *Advances in Engineering Software*, 23 pp.89-96.

Rock, S.J., Wozny, M.J. (1991) 'A flexible file format for solid freeform fabrication', Solid Freeform Fabrication Proceedings, Austin, TX, 1991, pp.1-12.

Rock, S.J., Wozny, M.J. (1992) 'Generating Topological Information from a Bucket of Facets', *proc., Solid Freeform Fabrication Symposium*, University of Texas at Austin, Austin, Texas, August 3-5,1992, pp. 251-258.

Rosochowski, A., Matuszak, A. (2000) 'Rapid tooling: the state of the art', *Journal materials processing technology* 106 (2000) 191-198.

Sabourin, E., Houser, S.A., Bohn, J.H. (1997) 'Accurate exterior, fast interior layered manufacturing', *Rapid Prototyping Journal*, Vol.3 No.2 1997 pp.44-52.

Sarkar, B., Menq, CH. (1991) 'Smooth surface approximation and reverse engineering', *Computer-Aided Design*, 23 (9), 1991, pp. 623–628.

SDRC (1999) 'I-DEAS Master Series™ 7. student guide', Structural Dynamic Research Corporation, Milford, OH, 1999.

Schoene, C., Hoffmann, J. (1997) 'Reverse engineering based on multi-axis digitized data', *International conference on manufacturing automation (ICNA '97)*, 1, pp.909-914

Sheng, X., Hirsch, B.E. (1992) 'Triangulation of trimmed surfaces in parametric space', *Computer-Aided Design*, Vol24, No.8, pp.437-444.

Sheng, X., Meier, I.R. (1995) 'Generating Topological Structures for Surface Models', *IEEE Computer Graphics & Applications*, Vol. 15, No. 6, pp. 35-41.

Spatial (2000) 'ACIS 6.0 technical overview', Spatial Technology Inc.
Stratasys (2000) 'Catalyst™ 1.2 release notes', Stratasys Inc.

Suh, Y., Wozny, M.J. (1994) 'Adaptive Slicing of Solid Freeform Fabrication Processes', *proc., Solid Freeform Fabrication Symposium*, University of Texas at Austin, Austin, Texas, August 8-10, 1994, pp.404-411.

Sun, W., Bradley, C., Zhang, Y.F., Loh, H.T. (2001) 'Cloud data modelling employing a unified, non-redundant triangular mesh', *Computer-Aided Design*, Vol. 33, No. 2, pp. 183-193.

Tata, K., Fadel, G., Bagchi, A., Aziz, N. (1998) 'Efficient slicing for layered manufacturing', *Rapid prototyping journal*, Vol 4, No.4 1998 pp.151-167.

Vafaeesefat, A. (1998) 'Operation planning for sculpture surface machining', Ph.D Dissertation of University of Windsor.

Vail, N.K., Wike, V.W., Bieder, H., Juneman, G. (1996) 'Interfacing reverse engineering data to rapid prototyping ', *proc., Solid Freeform Fabrication Symposium*, University of Texas at Austin, Austin, Texas, 1996, pp. 481-490.

Várady, T., Martin, R.R., Cox, J. (1997) 'Reverse engineering of geometric models – an introduction', *Computer-Aided Design*, Vol.29, No.4, pp.255-268.

West, A.P., Sambu, S.P., Rosen, D.W. (2001) 'A process planning method for improving build performance in stereolithography' *Computer-Aided Design* 33 (2001) pp. 65–79.

Wiedemann, B., Jantzen, H.A. (1999) 'Strategies and applications for rapid product and process development in Daimler-Benz AG' *Computer in industry* 39 (1999) pp.11-25.

Wozny, M.J. (1992) 'System issues in solid freeform fabrication', *Solid freeform fabrication symposium 1992*, H.L. Marcus et al., eds., University of Texas, Austin, August 1992, pp.1-15.

Yan X., Gu, P. (1996) 'A review of rapid prototyping technologies and systems', *Computer-aided Design*, Vol.28, No.4 pp307-318.

Yang, M., Lee, E. (1999) 'Segmentation of measured point data using a parametric quadric surface approximation' *Computer-Aided Design* 31 (1999) pp. 449–457.

Yau, H.T. (1997) 'Reverse engineering of engine intake ports by digitization and surface approximation', *Int. J. Tools Manufact.*, Vol.3, No.6, pp.855-871.

APPENDIX A

IMAGEWARE SURFACER SCRIPT FILE USED IN DIRECT CONTOUR GENERATION

```
##################################
## open file
##################################
#idm_execute("IDM_OPEN_DATA");

#set constant
lay_num = 40;      #number of cross-section layers

##################################
## Set environment
##################################
hide_all_curves();
hide_all_surfaces();
show_all_clouds();
fill_screen();
front_view();
cloud = clouds();

##################################
##Interactive parallel section
##################################
# Parallel section
printn("Parallel section: z axle");
idm_execute("IDM_POINT_CSECTION");

# Circular section
printn("Circular section: z axle");
idm_execute("IDM_PCLOUD_CIRCULAR_XSECT");

# Angular base feature
idm_execute("IDM_PDEVIATION");

cloud = clouds();
ang_num = cloud_num_data (cloud[0]);
cir_num = cloud_num_data (cloud[1]);
para_num = cloud_num_data (cloud[2]);
cir_box = cloud_bbox ( cloud[1]);
cir_min = cir_box[0];     #extract the minimum values of x, y, and z
cir_min_z = cir_min[2];  #extract the minimum values of z
cir_max = cir_box[1];    #extract the maximum values of x, y, and z
cir_max_z = cir_max[2]; #extract the maximum values of z
```

```
 # define circular section array
 # Override the default $max_vector_size temporarily
   nsave = $max_vector_size;
   $max_vector_size = para_num;
   para_points = vector (para_num);
   ang_points = vector (ang_num);
   cir_points = vector (cir_num);
   $max_vector_size = nsave;

 # assign the data cloud to array
 for(i= 0;i LT cir_num; i++){
       cir_points[i] = get_point(cloud[1],i+1);
       printn(i,"/",cir_num);
 }

 for(i= 0;i LT ang_num; i++){
       ang_points[i] = get_point(cloud[0],i+1);
       printn(i,"/",ang_num);
 }

 #for(i= 0;i LT para_num; i++){
 #     para_points[i] = get_point(cloud[2],i+1);
 #}

 # screen output
 printn(ang_num ,"  " ,cir_num," ", cir_box);
 # cross-section layer thickness
 height = (cir_max_z - cir_min_z)/lay_num;

 printn(height);

 lay_num =lay_num +1;

 n = vector(lay_num);     #circular point number in each layer
 n1 = vector(lay_num);    #angular point number in each layer
 cloud1 = vector(lay_num);
 cloud2 = vector(lay_num);
 flag = vector(lay_num);  # determine subregion
 a=0;
 a1=0;

 list = vector(1,cloud[2]);

 cir_min_x = cir_box[0][0];
 cir_max_x = cir_box[1][0];
 cir_min_y = cir_box[0][1];
 cir_max_y = cir_box[1][1];

 for(i= 0;i LT lay_num; i++){

       h= cir_min_z + i* height;      # Start layer height
       h1 = cir_min_z + (i+1)* height
       #printn(h);
       num = 0;
       num1 = 0;
```

```
        # Sorting circular data cloud according z height
        for(j= 0;j LT cir_num; j++){
              if ( cir_points[j][2] GE h AND cir_points[j][2] LT h1){
              #if ( cloud[1][j][2] GE h AND cloud[1][j][2] LT h1){
              num+=1;
              }
        }
        n[i]= num;

        # Sorting angular data cloud according z height
        for(j= 0;j LT ang_num; j++){
                    if ( ang_points[j][2] GE h AND ang_points[j][2] LT h1){
                    #if ( cloud[0][j][2] GE h AND cloud[0][j][2] LT h1){
                    num1+=1;
                    }
              }
        n1[i]= num1;

        #Calculate point extraction ratio
        ratio = (num1+0.01)/(num+0.01); # 0.01 used to avoid that divider is
"0"

        # determin flag according to ratio
        if (ratio GT 0.1){
              flag[i]=1;
        }else{
              flag[i]=0;
        }
}

#using extract_cloud_bbox to break the specified point clouds into multiple
#clouds based on spatial neighborhoods and a specified distance threshold.
#Preparing for curve fitting
for(i= 0;i LT lay_num; i++){

      h= cir_min_z + i* height;
      h1 = cir_min_z + (i+1)* height

      # extract points from boundary boxs
      b_box = vector
(2,vector(3,cir_min_x,cir_min_y,h),vector(3,cir_max_x,cir_max_y,h1));
      cloud1[i] = extract_cloud_bbox ( list, b_box);
}

for(i= 0;i LT lay_num; i++){

      h= cir_min_z + i* height;
      h1 = cir_min_z + (i+1)* height
      cloud2[i] = extract_distinct_clouds(cloud1[i],10);
}

# cross-section number
len = length(cloud2);
printn("cross-section cloud number is ",len);

#corss-section point number
cross_section_num = vector (len);
```

```
 curve = vector(len+1);

 cross_section = vector (len);
 err = vector(len);
 cur_keep = vector(len);
 tolerance = 0.1;
 j=1 ;
 cloud = clouds();

 intercur_num = 0;
 #Creates a B-spline interpolating curve of a 3 degree from the point cloud
 for(i= 0;i LT len; i++){
    if (length(cloud2[i]) GT 0){
       cross_section_num[i] = cloud_num_data(cloud2[i][0]);
       printn(i," ",cross_section_num[i]," ",flag[i]);
       if (cross_section_num[i] GT 4 AND flag[i] EQ 1){
             curve[j] = interpolate_curve_adv ( cloud2[i],$true,3);
             intercur_num++;
             #The following error information is returned in the order
 shown: maximum, average,
             #and standard deviation of positive normal error
             #err[i] = compute_curve_error_simp (curve[j],cross_section[i]);
             #if (err[i][1] LE tolerance){
             #     cur_keep[j]= curve[j];
                   j+=1;
             #}
       }
     }
 }

 # Circular section
 printn("Circular section: z axle NUMBER 1");
 idm_execute("IDM_PCLOUD_CIRCULAR_XSECT");
 cloud3 = vector(1)
 cloud = clouds();
 cloud3[0] = cloud[0]
 interpolate_curve_adv ( cloud3,0,3);

 ###################################
 #Extract control points and knots
 ###################################
 curve = curves();
 #printn(curve);
 intercur = vector(intercur_num);


 for(i= 0;i LT intercur_num; i++){
 intercur[i] = curve[i+1];
 }

 #align the start points of the curve
 change_start_point_using_spine ( intercur,curve[0]);

 printn(intercur);
 ctl_p_n = 20;
 #Reparametrizing them uniformly
 reparametrize_curve(intercur,1,intercur[0],ctl_p_n,0,0);
```

```
#declare control point array
control_point_x = vector(intercur_num);
control_point_y = vector(intercur_num);
control_point_z = vector(intercur_num);
knots = vector(intercur_num);
for(i= 0;i LT intercur_num; i++){
      control_point_x[i] = get_bspl_xcoefi ( intercur[i]) ;
      control_point_y[i] = get_bspl_ycoefi ( intercur[i]) ;
      control_point_z[i] = get_bspl_zcoefi ( intercur[i]) ;
      #Get knots from curves
      knots[i] = get_bspl_knots ( intercur[i])
}

#Define cloud5
cloud5 = vector(intercur_num);
for(i= 0;i LT intercur_num; i++){
  cloud5[i] = vector(ctl_p_n);
}

printn("Write control points and knots to file");
#writing a file including control points and knots
file_handle = open("E:/wu/controlpoints1.dat","w")
write(file_handle,intercur_num," ","CURVES NURMBER","\n");
for(i= 0;i LT intercur_num; i++){
  write(file_handle,length(cloud5[i])," ","CONTROL POINTS","
(",i,"/",intercur_num,")\n");
  for(j= 0;j LT ctl_p_n; j++){
           write(file_handle,control_point_x[i][j],"
",control_point_y[i][j]," ",control_point_z[i][j],"\n");
  }
  write(file_handle,length(knots[i])," ","KNOTS","
(",i,"/",intercur_num,")\n");
    for(j= 0;j LT length(knots[i]); j++){
        write(file_handle,knots[i][j],"\n");
  }
}
close(file_handle);
printn("OK! END! ");
```

APPENDIX B

OUTPUT FILE EXAMPLE OF DIRECT CONTOUR GENERATION METHOD

```
6 CURVES NURMBER
20 CONTROL POINTS (0/6)
496.694 -768.673 82.9518
497.842 -772.318 82.9518
501.343 -779.612 82.9518
514.929 -778.780 82.9518
524.958 -777.324 82.9518
534.730 -770.110 82.9518
540.915 -760.939 82.9518
542.801 -749.360 82.9518
542.321 -738.112 82.9518
539.531 -727.366 82.9518
533.942 -716.852 82.9518
524.255 -711.028 82.9518
512.673 -711.028 82.9518
501.023 -713.253 82.9518
495.573 -723.837 82.9518
494.448 -735.182 82.9518
494.448 -746.278 82.9518
494.448 -757.743 82.9518
495.546 -765.028 82.9518
496.694 -768.673 82.9518
24 KNOTS (0/6)
0
0
0
0
.058824
.117647
.176471
.235294
.294118
.352941
.411765
.470588
.529412
.588235
.647059
.705882
.764706
.823529
.882353
.941176
1
1
1
1
...
```

APPENDIX C

EXAMPLE OF CONTOUR FILE (*.SGM) CREATED IN CATALYST

```
Slice V19
STEP 0.01
HEIGHT 4.76
G (0 DEFAULT)
G (1 BaseFast)
G (2 BaseTop)
G (3 Part)
G (4 PartFast)
G (5 PartSmall)
G (6 PartMating)
G (7 SupportFace)
G (8 Support)
G (9 SupportBottomSmall)
G (10 SupportBottom)
G (11 SupportSparse)
G (12 SupportPerf1)
G (13 SupportPerf2)
G (14 SupportSmall)
G (15 PartXHatch)
G (16 PartXtended)
FILE(0,Drawn,DRAWN)
FILE(1,C:/model/autoshift.stl,id)
Z -0.04 0.01 F
O 1
1
1
86
1.8110842 2.761445
1.764616 2.8079128
1.7127316 2.8487723
1.6561563 2.8827276
1.596136 2.9090271
1.5337015 2.9277239
1.4697384 2.9390926
1.4050782 2.943377
1.3404919 2.9407699
1.2777747 2.9316759
1.2443267 2.9240308
1.2140568 2.9153595
1.1534171 2.8927336
1.0952947 2.863543
1.0551219 2.8372724
1.0488893 2.8578835
1.0452029 2.8658817
1.0413464 2.8727624
1.036456 2.8789511
....
```

VITA AUCTORIS

NAME:	Tingzhou Wu
PLACE OF BIRTH:	Shanghai, China
YEAR IF BIRTH:	1969
EDUCATION:	Jiangsu Institute of Technology, Zhenjiang, Jiangsu, China
	1987 – 1991 B.E.Sc
	University of Windsor, Windsor, Ontario, Canada
	1999 – 2001 M.A.Sc